AMIGA WORLD

N.º 7 - Febrero 1990 - Publicación de IDG Communications - 500 Ptas.

PROGRAMACION

- Ordenación en Basic
- Animación del puntero
- Trepa con A-REXX

BANCO de PRUEBAS

- MiniGen
- Dominio Público
- Libros

SECCION COMMODORE WORLD

- Interface RS-232



ISABEL TAPIAS



Director:
Juan Manuel Urraca

Redacción y Dpto. Técnico:
Aitor Urraca
Francisco Javier Rodríguez

Diseño y maquetación:
Miguel Angel Hermosell

Dpto. publicidad:
Gloria Montalvo (Madrid)
Magda Zabala (Barcelona)

Distribución y suscripciones:
Fernando Rodríguez (dirección)
Angel Rodríguez
Basilio Rodríguez (suscripciones)

Edita:
CW COMMUNICATIONS

Director general:
Francisco Zabala

Oficina principal:
C/. Rafael Calvo, 18 - 4.° B
Tel. (91) 319 40 14
Fax: 319 61 04
28010 Madrid

Departamento de suscripciones:
C/. Zurbano, 39 - bajo dcha.
Tel. (91) 308 19 30
28010 Madrid

Delegación en Barcelona:
C/. Bertrán, 18-20, 3.° - 4.ª
Tels. (93) 212 73 45 - 212 88 48
08023 Barcelona

C.I.F. A-28-735389

El P.V.P. para Ceuta, Melilla y Canarias, incluido servicio aéreo, es de 500 ptas., sin I.V.A.

Distribuidor:
SGEL
Avda. Valdelaparra, s/n.
Pol. Ind. de Alcobendas
Madrid

Imprime:
OMNIA, I.G.
Mantuano, 27
28002 Madrid

Fotocomposición:
ANDUEZA, S. A.
San Romualdo, 26, 7.ª planta
28037 Madrid

Fotomecánica:
RODACOLOR

Depósito legal: M. 9064-1989

E D I T O R I A L

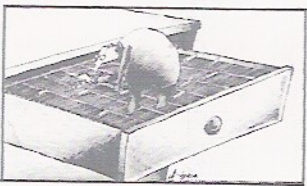
Dentro de los aspectos más prácticos de un equipo informático, la programación ocupa un puesto relevante por su condición de nexo total entre el usuario y la máquina. En el Amiga no podía ocurrir de otra manera. Aunque somos conscientes de la gran cantidad de software existente en el mercado, y de que muchos usuarios no desarrollan programas propios, hemos considerado importante dedicar un número de nuestra publicación a la programación.

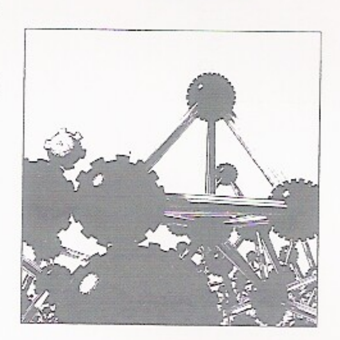
Para que usted pueda encontrar variedad, se han publicado artículos de diversos temas. El más didáctico y completo es probablemente la ordenación en basic. Con sus diferentes métodos, explicaciones y ejemplos prácticos. Pero también se deben destacar los artículos dedicados al puntero. Ese elemento de presencia permanente en la pantalla del Amiga, consigue en algunos programas llamar la atención por su calidad gráfica o por un aspecto poco conocido, la animación. Realmente es más sencillo de lo que parece. Por medio de los programas de animación y edición que presentamos, podrá dar vida a sus punteros sin dificultad.

Entre las novedades que presentará el próximo sistema operativo, Workbench 1.4 que aún no está disponible en versión definitiva, se encuentra A-REXX. Poco a poco se está implantando en el desarrollo de las más diversas utilidades e incluso juegos. La información de esta potente herramienta para los programadores es todavía precaria, pero uno de nuestros colaboradores ha estudiado el tema y nos lo presenta de una forma didáctica y práctica.

En cuanto al mercado de Amiga, sigue su lenta línea ascendente, potenciada en el sector profesional y consolidada en aspecto de entretenimiento. En el próximo número publicaremos una demostración gráfica de lo que es capaz Amiga en el terreno del CAD/CAM, a nivel profesional.

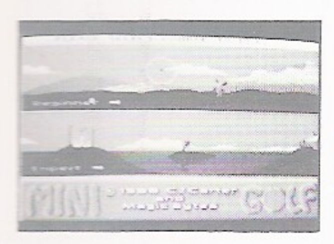
AMIGA WORLD es una publicación de IDG Communications, el mayor grupo editorial del mundo en el ámbito informático. IDG Communications edita más de 90 publicaciones relacionadas con la informática en más de 34 países. Catorce millones de personas leen una o más de estas publicaciones cada mes. Las publicaciones del grupo contribuyen al Servicio Internacional de Noticias de IDG que ofrece las últimas noticias nacionales e internacionales sobre el mundo de la informática. Las publicaciones de IDG Communications incluyen: **ARABIA SAUDI:** Arabian Computer News. **ARGENTINA:** Computerworld Argentina. **ASIA:** Communications World; Computerworld Hong Kong; Computerworld South East Asia; PC Review. **AUSTRALIA:** Computerworld Australia; Communications World: Australian PC World; Australian Macworld. **AUSTRIA:** Computerwelt Oesterreich. **BRASIL:** DataNews; PC Mundo; Micro Mundo. **CANADA:**Computer Data. **CHILE:** Informática; Computación Personal. **COREA DEL SUR:** Computerworld Korea; PC World Korea. **DINAMARCA:** Computerworld Danmarck; PC World Danmark; CAD/CAM World. **ESPAÑA:** Computerworld España; PC World España; Commodore World; Comunicaciones World; CIM World. **ESTADOS UNIDOS:** Amiga World; CD-ROM Review; CIO; Computer Currents; Computerworld; Digital News; Federal Computer Week; 80 Micro; FOCUS Publications; InCider; Infoworld; Macintosh Today; Macworld; Computer + Software News (Micro Marketworld/Lebhar-Friedman); Network World; PC Letter; PC World; Portable Computer Review; Publish!; PC Resource; Run. **FINLANDIA:** Mikro; Tietiviikko. **FRANCIA:** Le Monde Informatique; Distributique; InfoPC; Telecoms International. **GRECIA:** Computer Age. **HOLANDA:** Computerworld Netherlands; PC World Benelux. **HUNGRIA:** Computerworld SZT; PC Microvilag. **INDIA:** Dataquest; PC World India. **ISRAEL:** People & Computers Weekly; SBM Monthly. **ITALIA:** Computerworld Italia. **JAPON:** Computerworld Japan; Semicon News. **MEXICO:** Computerworld Mexico; PC Journal. **NORUEGA:** Computerworld Norge; PC World Norge. **NUEVA ZELANDA:** Computerworld New Zeland. **REINO UNIDO:** Computer News; ICL Today; LOTUS; PC Business World. **REPUBLICA FEDERAL ALEMANA:** Computerwoche; Information Management; PC Welt; PCWoche; Run/Run Specials. **REPUBLICA POPULAR CHINA:** China Computerworld; China Computerworld Monthly. **SUECIA:** Computer Sweden; Mirkro Datorn; Svenska PC World. **SUIZA:** Computerworld Schweiz. **VENEZUELA:** Computerworld Venezuela.





S U M A R I O

44 JUEGA CON TU AMIGA



Nuestra famosa sección de comentarios de juegos. Algunos nacionales, la mayoría buenos y todos interesantes.

- War in Middle Earth
- The Games Summer Edition
- Mission Elevator
- Beach Volley
- Mini Golf
- Dragon Spirit
- Astaroth
- Beam

51 BOLETINES DE PEDIDOS AMIGA

Para solicitar los números atrasados o los discos editados por nuestra publicación.

52 BANCO DE PRUEBAS

Pruebas y novedades sobre productos del mercado nacional y extranjero. Lo más interesante de este número.

- MiniGen
- Amiga ROM Kernel Reference Manual
- Amiga Hardware Reference
- Dominio público

58 DIRECTORIO

59 BOLETINES DE PEDIDO COMMODORE

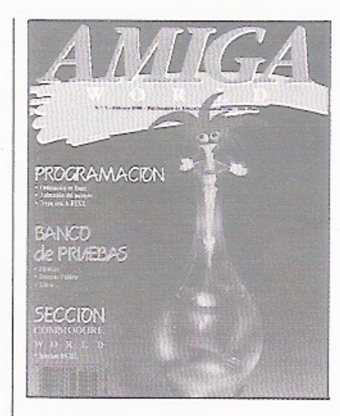
Para solicitar las revistas o discos editados en la revista Commodore World.

sección Commodore WORLD

60

En estas páginas podrá encontrar la más diversa información y programas sobre los modelos de ocho bits de la marca Commodore.

- Interface RS-232 para Commodore 64 o Plus-4, con conexión a PC.



PORTADA: ISABEL TAPIAS

CURSOS EN VIDEO

La empresa barcelonesa Prosystem está comercializando desde el mes pasado, cursos de diversos programas, en video. Los programas que se tratan son los siguientes: Sculpt 4D, Provideo, VideoEffects 3D y Deluxe Paint III.

La duración de las cintas es como máximo de dos horas. El precio está alrededor de las 9.900 pesetas, que en ningún caso se superan. Además, están garantizados y son una importante novedad en el mercado nacional. Pero lo más importante para el usuario, además de la calidad de los cursos, es el servicio Hot-Line de un mes que esta compañía ofrece a sus clientes. Una vez recibido el curso en video y enviada la correspondiente garantía, el usuario tiene derecho a cuantas consultas telefónicas precise durante todo un mes. Servicio con calidad profesional.

Por otra parte, Prosystem está comercializando en exclusiva los Genlocks de la firma alemana Electronic Design. En sus dos versiones Amateur y Profesional, disponen de controles como Fade, Mode, luminancia, contraste, color e invert. El modelo Profesional tiene calidad S-VHS con entradas y salidas separadas para esta señal, además de la entrada y salida de video compuesto como en el amateur. Los precios son 49.900 y 94.900 para el Amateur y Profesional, respectivamente. ■

TXIPARADE

Cada mes publicaremos los diez juegos más votados por los seguidores del programa Teletxip de la televisión vasca. En el mes de enero han correspondido a:

1. INDIANA JONES
2. LOS INTOCABLES
3. SHINOBI
4. DANKO RED HEAT
5. DOUBLE DRAGON
6. TINTIN EN LA LUNA
7. OPERATION THUNDERBOLT
8. LA AVENTURA ORIGINAL
9. DYNAMITE DUX
10. BALLISTIX

CONCURSO COMMODORE DE DIBUJO CON AMIGA

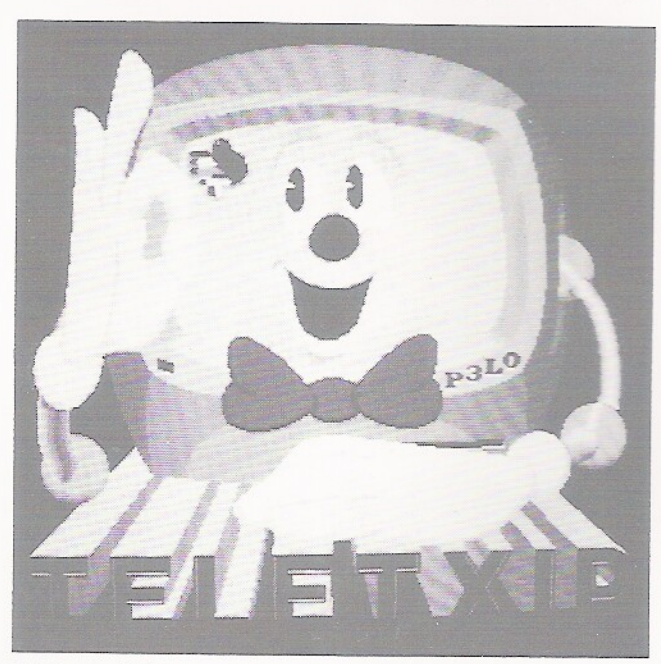
Commodore, S.A. ha organizado un concurso de dibujo para usuarios de Amiga. Existen dos categorías, aficionados y profesionales. Además de los tres primeros premios por cada categoría, se concederán tres accésit en los que participa la revista Amiga World con 6 Suscripciones de revista + disco. Para más información respecto al presente concurso diríjanse a la compañía Commodore y soliciten las bases.

La fecha límite de entrega de trabajos será el próximo día 30 de abril.

Por nuestra parte, publicaremos las incidencias y evolución del concurso, así como la lista de premiados en su momento. ■

DELUXEVIDEO III

Electronic Arts nos anuncia la disponibilidad de la nueva versión de su programa DeluxeVideo. La nueva versión de este programa para «desktop video», ofrece calidad y efectos profesionales, con un interface de usuario más intuitivo. Entre otras características, soporta todos los modos de resolución gráfica de Amiga, incluyendo HAM. Dispone de animación en alta velocidad, salida MIDI e IFF para sonido y ficheros de música, etc. Se puede combinar con las brochas animadas de DeluxePaint III y efectos sonoros digitalizados, para crear sofisticadas presentaciones audiovisuales. Por medio de sus controles externos se puede grabar el video creado, a través de Genlock o directamente en disco para ser visualizado posteriormente. ■



Phirulo, especie de monitor parlante, es la mascota de Teletxip.

POPULOUS Y DELUXE PAINT III, PREMIADOS

El pasado 18 de diciembre fueron premiados en una cena de la industria norteamericana, estos dos productos relacionados con el Commodore Amiga. Electronic Arts consigue de esta manera un reconocimiento a su prolongada y pionera labor en el desarrollo de software para Amiga, incluso desde antes del lanzamiento de la máquina al mercado.

Populous, juego que ya comentamos en el número cuatro de Amiga World, ha sido premiado como el mejor del año 1989. En la redacción coincidimos con el veredicto.

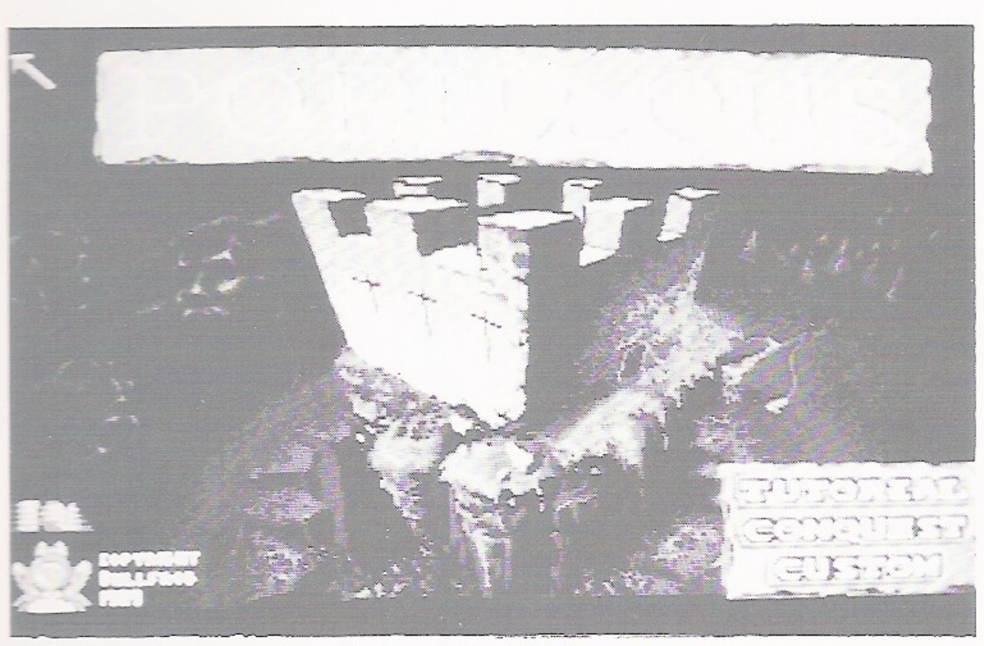
Deluxe Paint III también ha sido premiado en el mismo evento, como el producto que mejor aprovecha las características del Amiga.

Mark Lewis, director general de Electronic Arts, señalaba la importancia del soporte que su compañía ha dado al Amiga desde los comienzos.

Aprovechamos la ocasión para felicitar por estos premios, tanto a la compañía americana como a su distribuidor en nuestro país, Dro Soft. ■

PROXIMO PROGRAMA DE TELETXIP

El próximo 7 de febrero, el programa de la Televisión Vasca (ETB1) Teletxip, emitirá un interesante capítulo en el que se darán a conocer las aplicaciones que en DISNEYLANDIA, se les ha dado a



Este juego Populous fue premiado al igual que Deluxe Paint III.

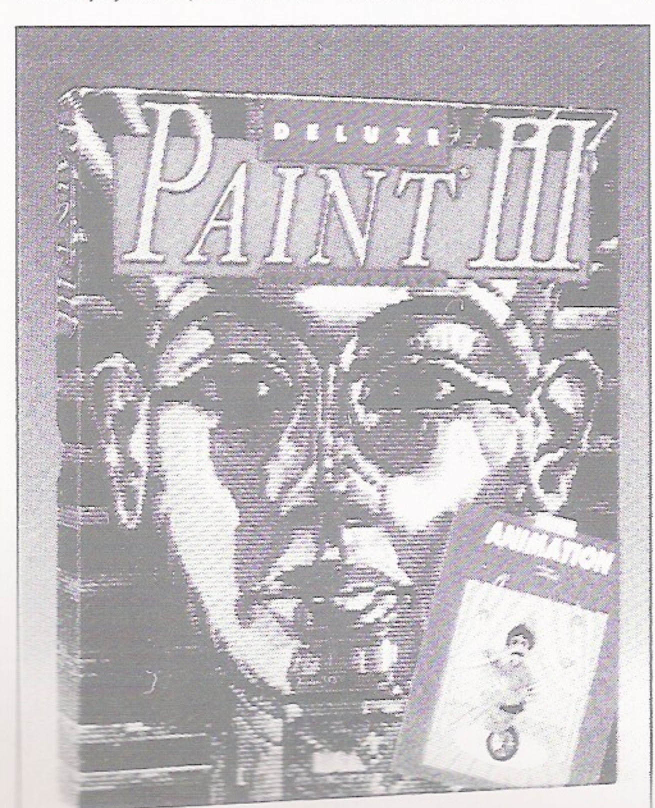
los ordenadores. También un reportaje sobre la cámara aérea que se utiliza para tomar las jugadas de baloncesto en la NBA.

Asimismo durante el programa se repartirán los regalos del Concurso de Dibujo y Ejercicio Basic. Entre los regalos hay un Commodore Amiga 500 y también libros, juegos etc. ■

VES ONE SOLUCION COMPLETA EN VIDEO

Videotechnik es un nuevo establecimiento especializado en video con Amiga, que acaba de abrir sus puertas en Madrid. Y lo ha hecho con una interesante noticia, la presentación de la unidad VES ONE. Este producto desarrollado por la firma alemana Videocomp, está distribuido en exclusiva por Videotechnik. Está diseñado para crear efectos profesionales tanto con el Amiga 500 como con el 2000.

VES ONE es titulación, efectos y proceso de señales de video, todo en uno. Todas las operaciones pueden ser realizadas con un simple click del ratón. Tiene un funcionamiento profesional, facilidad de manejo y no se precisan conocimientos informáticos. Los componentes del equipo son: Genlock VHS y S-VHS/Hi8, Digitalizador Digi View Gold 3.0 con filtro RGB electrónico, procesador de video y mezclador de video. ■



NUEVOS PRODUCTOS CIMEX-MICROTECK

Entre los muchos productos que comercializa la empresa de Barcelona, acaba de presentar para Amiga, Handy Scanner y un digitalizador de sonido. El scanner manual está fabricado por la compañía alemana Cameron y el digitalizador, por la inglesa Trilogic. Los precios son de 12.000 pts para el digitalizador y 50.000 pts para el scanner. Este último incluye el interface para Amiga 500 y 2000. En el próximo número de nuestra revista publicaremos un banco de pruebas de ambos productos, que esperamos sean de su interés.

Por otra parte Microteck, perteneciente al grupo Cimex, está fabricando ampliaciones de memoria para Amiga 500. Las diferentes configuraciones van desde 0,5 mb hasta 2 mb. Los precios oscilan entre las 28.900 y las 46.900 pts. También está fabricando un MIDI para los modelos 500 y 2000 al precio de 11.900 ptas.

Esperamos comentar en breve los diversos productos de este fabricante nacional. ■

DEMOSTRACION DEL X-CAD

La compañía Commodore está organizando unas demostraciones profesionales del programa X-CAD. Este programa es conocido por su especial aprovechamiento de las posibilidades del Amiga para el desarrollo de trabajos CAD/CAM. Será presentado por un experto en los locales de la propia compañía Commodore, a partir del próximo día 7 de febrero. Para solicitar la asistencia, deben ponerse en contacto con la señorita Pilar, en los teléfonos de Commodore, S. A., en Madrid. ■

ORDENACION

Por Alvaro Ibáñez

Prácticamente cualquier aplicación, desde una simple agenda hasta una base de datos, necesita realizar algún tipo de ordenación con variables numéricas o alfanuméricas. Cuando llega la hora de programar una aplicación en la que se emplee una rutina de ordenación (Sort, en inglés), es necesario disponer del sistema más rápido posible, dado que la velocidad del programa suele ser un factor clave. Por esta razón, este artículo presentará algunos de los sistemas más conocidos, indicando sus características, forma de funcionamiento y verdadera utilidad.

Se ha elegido el Basic por ser el lenguaje más popular y estar a disposición de todos los usuarios de Amiga. Cualquiera que tenga conocimientos de C o Ensamblador puede convertir fácilmente las rutinas aquí presentadas, sobre todo porque se encuentran comentadas y explicadas en detalle.

Usuarios principiantes

Uno de los primeros problemas de programación que todo usuario novato debería plantearse es el de realizar una ordenación de números o palabras. Es un ejercicio muy instructivo que, al cabo de algunas horas de razonamiento, puede proporcionar buenos frutos. Un buen método para desarrollar una rutina de este tipo consiste en seguir un sistema «manual», utilizando cartulinas con números y razonando cuáles son los pasos que sigue mentalmente una persona para ordenarlos.

Al programar en Basic, la idea es partir de una matriz numérica en la que se encuentran varios números de forma desordenada. Los datos se pueden leer desde el teclado, sentencias DATA o generarlos de forma aleatoria. El ordenador debe encargarse de imprimir la serie, llamar a la rutina de ordenación (mediante GOSUB o CALL), e imprimir el resultado: la lista de números ordenada de menor a mayor.

No son necesarios grandes conocimientos para llevar a cabo este sencillo ejercicio. Tan sólo saber cómo trabajan las matrices en Basic, un poco de intuición... y la rutina de ordenación estará lista. El resultado final seguramente será el método de ordenación conocido como Burbuja (Bubble Sort) o Selección (Select). Es lógico: son los más sencillos e intuitivos, aunque no los más eficaces, como se podrá comprobar a continuación.

Existen sistemas de ordenación muy variados que conviene hacer uso de ellos de la forma más apropiada en cada aplicación. Al mismo tiempo, resulta instructivo comprender la base de su funcionamiento.

Distintos métodos

Existen cinco métodos de ordenación que se cuentan entre los más populares. Estos métodos han ido evolucionando poco a poco desde los primeros tiempos de la informática, de modo que algunos son verdaderamente primitivos, como el método de la burbuja o el de selección. Otros, como el de Ordenación Rápida (Quick-Sort), hacen uso de recursividad, un sistema de programación relativamente moderno que no todos los intérpretes Basic admiten (Amiga Basic, por ejemplo, no).

Los listados 1 al 6 contienen los listados de las seis rutinas de ordenación, en forma de rutinas SUBs. De este modo pueden insertarse fácilmente en cualquier programa creado por el usuario, dado que no interfieren con las variables existentes.

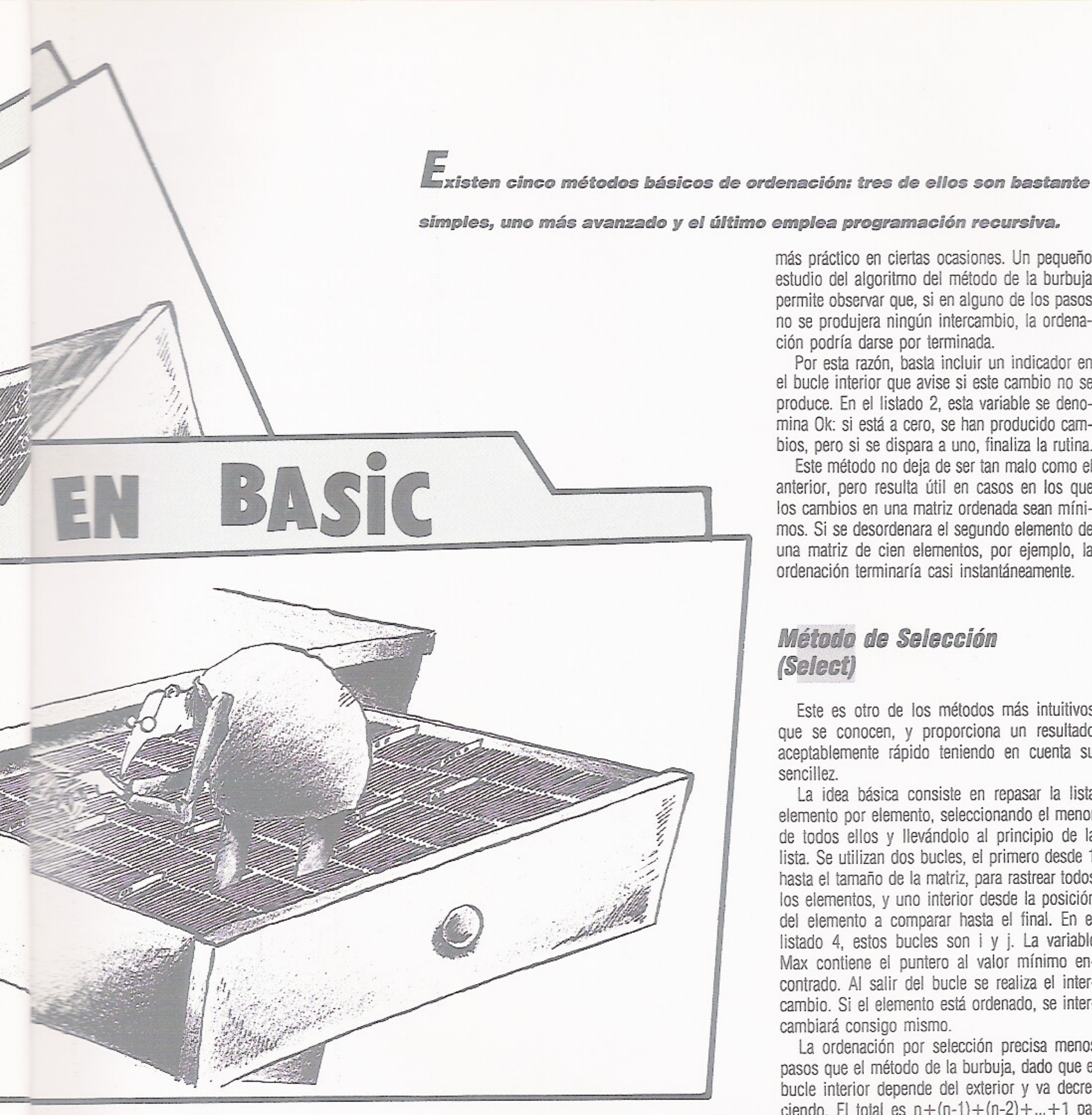
Los listados 7 y 8 contienen demostraciones que pueden servir a modo de ejemplo para conocer cómo emplear las rutinas. A continuación del listado 7 deben añadirse los listados 1 al 5, y a continuación del listado 8, el 6.

La sintaxis de las rutinas es en todos casos la misma:

```
CALL NombreRutina(Matriz(),Primero,Ultimo)
```

Matriz() es la matriz numérica que se desea ordenar y Primero y Ultimo indican cuáles son los elementos inicial y final. Para ordenar toda la matriz se deben pasar los valores 1 y el límite de la matriz. Al final del artículo se explicará cómo ampliar estas rutinas para realizar más

Existen cinco métodos básicos de ordenación: tres de ellos son bastante simples, uno más avanzado y el último emplea programación recursiva.



funciones, como ordenar matrices alfanuméricas o en sentido inverso.

Método de la Burbuja (Bubble Sort)

Este es el método de ordenación más popular, que resulta ser al mismo tiempo el más lento e ineficaz. Resulta muy sencillo de programar, y emplea muy pocas líneas de código. Sin embargo, los tiempos que se obtienen son verdaderamente ridículos.

Este método funciona del siguiente modo: Uno por uno, se va comparando cada elemento con todos los demás de la lista. En cada comparación, se intercambian los dos elementos si están desordenados (para colocar en primer lugar el más pequeño). De este modo, los elementos van subiendo y bajando a lo largo de la lista para colocarse cada uno en su lugar.

Este sistema es tremendamente ineficaz, dado que precisa aproximadamente n^2 pasos para completar la ordenación de n elementos, por lo que el tiempo de cálculo crece de forma exponencial. El listado 1 muestra la rutina BubbleSort.

Método de la Burbuja Mejorado (QuickBubble)

Aunque este sistema es prácticamente idéntico al anterior, introduce una mejora que lo hace más práctico en ciertas ocasiones. Un pequeño estudio del algoritmo del método de la burbuja permite observar que, si en alguno de los pasos no se produjera ningún intercambio, la ordenación podría darse por terminada.

Por esta razón, basta incluir un indicador en el bucle interior que avise si este cambio no se produce. En el listado 2, esta variable se denomina Ok: si está a cero, se han producido cambios, pero si se dispara a uno, finaliza la rutina.

Este método no deja de ser tan malo como el anterior, pero resulta útil en casos en los que los cambios en una matriz ordenada sean mínimos. Si se desordenara el segundo elemento de una matriz de cien elementos, por ejemplo, la ordenación terminaría casi instantáneamente.

Método de Selección (Select)

Este es otro de los métodos más intuitivos que se conocen, y proporciona un resultado aceptablemente rápido teniendo en cuenta su sencillez.

La idea básica consiste en repasar la lista elemento por elemento, seleccionando el menor de todos ellos y llevándolo al principio de la lista. Se utilizan dos bucles, el primero desde 1 hasta el tamaño de la matriz, para rastrear todos los elementos, y uno interior desde la posición del elemento a comparar hasta el final. En el listado 4, estos bucles son i y j. La variable Max contiene el puntero al valor mínimo encontrado. Al salir del bucle se realiza el intercambio. Si el elemento está ordenado, se intercambiará consigo mismo.

La ordenación por selección precisa menos pasos que el método de la burbuja, dado que el bucle interior depende del exterior y va decreciendo. El total es $n+(n-1)+(n-2)+...+1$ pasos, es decir, $(n^2-n)/2$.

Método de Inserción (Insert)

Este es último de los métodos denominados «simples», y es el más empleado para listas no demasiado grandes cuando se necesita que la rutina no sea demasiado complicada. El método de inserción es similar al que se emplea cuando se ordenan las cartas de una baraja: insertando cartas en su lugar entre las que ya están ordenadas.

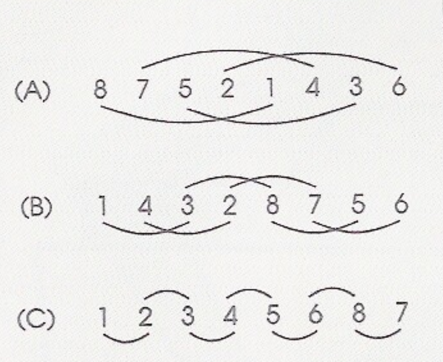
Esta rutina (Listado 5) cuenta con un bucle desde primer elemento más uno hasta el final de la matriz. Cada elemento se va comparando

Los métodos Shell y QuickSort mejoran su rendimiento cuantos más elementos compongan la lista.



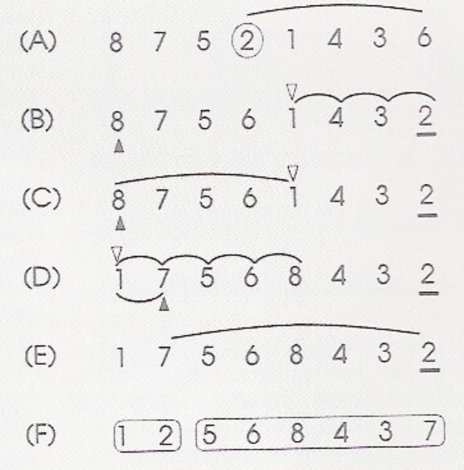
Ordenación SHELL

El método Shell ordena la lista realizando comparaciones entre elementos separados por un intervalo fijo, que va disminuyendo progresivamente.
(A) Se comienza con un intervalo igual a la mitad del número de elementos, en este caso 4. Las parejas se ordenan independientemente.
(B) El intervalo se va reduciendo, dividiéndolo progresivamente por dos. En este segundo paso, el intervalo es 4/2, es decir, 2.
(C) Finalmente, el intervalo llega a 1.
(D) La lista queda ordenada tras el último paso.



CUADRO 2

Ordenación QUICKSORT



El método QuickSort ordena una lista de elementos seleccionando un punto al azar, agrupando los elementos menores y mayores a ambos lados del pivote y realizando una llamada recursiva para ordenar los dos grupos.
(A) Se elige al azar un elemento pivote (en este ejemplo, el número 2). Este valor, que será el utilizado para las comparaciones, se intercambia con el último elemento de la lista. Más adelante se colocará en el lugar que corresponda.
(B) Dos punteros (I y J, representados por una flecha negra y otra blanca en el gráfico) recorren la lista de izquierda a derecha y de derecha a izquierda. El primero se detendrá cuando el elemento al que apunta sea mayor que el pivote, y el segundo cuando sea menor que el pivote. En este caso, al llegar a los valores 8 y 1.
(C) Los dos elementos se intercambian.
(D) La búsqueda con los punteros I y J se repite, hasta que J sea menor que I.
(E) Terminado el agrupamiento, se intercambia el pivote (2) por el valor de I. De este modo el pivote queda en su lugar.
(F) El resultado son dos sub-listas, de 1 a J y de I al último elemento. La primera es globalmente «menor» que la segunda. Ambas listas se ordenan respectivamente mediante llamadas recursivas a QuickSort. Cuando las listas tienen uno o dos elementos, QuickSort finaliza.

con los anteriores hasta que alguno es igual o mayor. En ese momento se inserta el elemento en esa posición. ¿Qué sucede con el resto de la lista que ya está ordenada y hay que desplazar? El mismo bucle de búsqueda se emplea para ir desplazando (insertando) todos los elementos uno por uno.

El numero de pasos empleados por el método de inserción es, en el peor de los casos, $(n^2-n)/2$. Si la matriz está casi ordenada, los pasos serán muchos menos y la velocidad mayor. Por esta razón, resulta mucho más eficiente que el método de la burbuja. La conclusión que se obtiene es que este método resulta especialmente útil cuando la matriz no se encuentra excesivamente desordenada, es decir, cuando los elementos a ordenar se encuentran próximos a su posición de destino.

Existe una variante denominada «Inserción Binaria» que aprovecha el hecho de que la primera parte de la lista va quedando ordenada, para encontrar rápidamente (mediante búsqueda binaria) la posición correspondiente al elemento que se está ordenando.

Método Shell

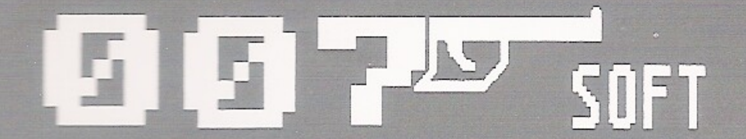
Cuando la cantidad de elementos de la matriz a ordenar es muy grande, se precisa métodos más avanzados y rápidos que los antes expuestos. Estos métodos proporcionan más velocidad aunque resultan menos intuitivos y algo más complicados de programar.

El método Shell es uno de los más rápidos y puede ser adaptado a cualquier intérprete Basic. Su nombre se debe a su inventor, D. L. Shell, y por lo tanto no tiene traducción (hay quien lo denomina erróneamente «Método de la Concha» o «Método del Cascarón»).

Este método se basa en ordenaciones sucesivas de la lista por parejas de números. Estas parejas están separadas por un intervalo (número de posiciones) fijo. Al ir disminuyendo este intervalo, la lista queda progresivamente ordenada. Cuando el intervalo llega a 1, la lista está ordenada.

El cuadro 1 muestra gráficamente este sistema de ordenación. En el ejemplo, la lista se compone de ocho números desordenados, aunque en la práctica pueden ser cientos o miles. Tras las sucesivas ordenaciones parciales, la lista queda totalmente ordenada. Debe observarse que hasta que una de las listas lista parcial no está totalmente ordenada por pares, no se pasa al siguiente intervalo.

En el programa (Listado 3), la variable Ancho es la que controla el intervalo. La variable Flag permite comprobar que la lista está correcta-



concesionario **MASTER DEALER** de Commodore

Especialistas en:

* Commodore AMIGA
* Atari ST

- *Rotulaciones en vídeo, animaciones y logotipos en 2D y 3D.*
- *Autoedición asequible a cualquier bolsillo.*
- *Digitalización y tratamiento de imágenes de vídeo, fotografía, etc.*
- *MIDI profesional a su alcance.*

		PRESUPUESTO MINIMO	PRESUPUESTO ACONSEJABLE	PRESUPUESTO MAXIMO
Amiga 500	95.000 ptas.	●		
Amiga 2000	225.000 ptas.			●
Monitor 1084	58.000 ptas.	●		●
Modulador TV, A-500	6.500 ptas.		●	
Modulador TV, A-2000	15.000 ptas.		●	
Exp. 512 K. A-500 No Commodore	25.000 ptas.		●	
Exp. 512 K. A-500 Commodore	33.000 ptas.	●		
Exp. 1.8 Megas A-500	75.000 ptas.		●	
Exp. 2 Megas A-2000 No Commodore	85.000 ptas.		●	
Exp. 2 Megas A-2000 Commodore	98.000 ptas.		●	
Exp. 8 Megas A-2000	280.000 ptas.			●
Disco Duro 20 Megas A-500	98.000 ptas.		●	
Disco Duro 20 Megas A-2000	115.000 ptas.		●	
Disco Duro 40 Megas A-2000	145.000 ptas.			●
Unidad 3 1/2 interna	23.000 ptas.			●
Unidad 3 1/2 externa No Commodore	25.000 ptas.		●	
Unidad 3 1/2 externa Commodore	35.000 ptas.			●
Genlock Minigen	35.000 ptas.	●		
Genl. NHS	70.000 ptas.		●	
Genl. SATV SVHS	115.500 ptas.		●	
Genl. SATV Professional	425.900 ptas.			●
Digitalizador Digi View	27.000 ptas.	●		
Digit. Superpic	175.000 ptas.			●
Separador de colores NHS	45.000 ptas.	●		
Separador de colores SATV	66.500 ptas.			●
Frame Buffer (2 Megas)	152.500 ptas.			●
Scanner Printecnik	109.900 ptas.		●	
Scanner Printecnik Professional	285.900 ptas.			●
Cable Euroconector-Amiga	5.500 ptas.		●	

Para demostraciones y consultas rogamos nos llamen antes, para concertar hora de visita con nuestro equipo de especialistas, que gustosamente atenderán sus dudas.

ALCALA, 211
28028 MADRID

TFO. (91) 256 14 30
FAX (91) 256 16 91

mente ordenada por pares antes de pasar al siguiente intervalo.

El método SHELL es sumamente eficaz, y necesita aproximadamente n1.2 pasos para ordenar la lista. Por esta razón, es mucho más recomendable para matrices con muchos elementos que los métodos anteriores.

Método QuickSort

El último de los métodos se denomina QuickSort (Ordenación Rápida) y fue creado por C. R. Hoare. Se trata del método más rápido de todos, y muchos (yo entre ellos) lo consideran el más elegante. Aunque existen distintas variaciones, la base del método es la recursividad. Se dice que un programa o rutina es recursivo cuando se «llama» a sí mismo. En este caso, QuickSort divide la lista de elementos en listas más pequeñas y se llama a sí mismo para ordenarlas, una y otra vez, recursivamente.

El intérprete Basic del Amiga es tan lamentable y malo que, entre sus muchos defectos se encuentra el de no permitir recursividad. El gracioso mensaje «Esa rutina ya se está utilizando» te dirige al manual, donde se especifica claramente que «La recursividad no está permitida». (Esto le hace a uno plantearse por qué de entre todo el software existente, el peor tiene que ser el que te regalan con la máquina).

De modo que esta rutina, tal y como está diseñada, no puede utilizarse en Amiga Basic. Por esta razón, se ha desarrollado bajo True BASIC, otro de los intérpretes Basic para el Amiga, que sí permite recursividad, aunque también tiene algunas incomodidades.

QuickSort parte de la lista desordenada y elige un elemento que se denomina «Pivote». Este punto puede ser el punto central o un punto aleatorio de la lista (este método se conoce genéricamente como «Montecarlo» en términos informáticos, y suele proporcionar mejores resultados). A continuación, agrupa en uno de los lados todos los elementos menores que el pivote, y a otro lado los mayores. El resultado son dos sublistas, que se ordenan llamando de forma recursiva a la rutina QuickSort.

El cuadro 2 muestra gráficamente y de forma práctica una ordenación mediante QuickSort.

QuickSort es el método más rápido de todos los conocidos, y algunas de sus variantes se utilizan «de fábrica» en muchos compiladores de C, en forma de función (qSort, generalmente) de forma que el usuario no necesita ni desarrollar esta rutina.

En el listado pueden apreciarse perfectamente todos los pasos que realiza QuickSort para ordenar la matriz. Se ha empleado una rutina



Resultados comparativos de las diversas rutinas

	10	25	50	100	250	500	1.000
Bubble	00.25"	00.75"	02.75"	11.13"	01.12"	04.55"	19.31"
QuickBubble	00.25"	00.75"	02.75"	11.25"	01.12"	04.55"	19.19"
Select	00.25"	00.63"	01.25"	04.13"	28.13"	01.52"	07.28"
Insert	00.25"	00.75"	00.75"	03.63"	20.63"	01.29"	05.33"
Shell	00.02"	00.25"	01.25"	02.13"	10.25"	26.02"	01.12"
QuickSort	00.06"	00.22"	00.52"	01.66"	08.32"	31.08"	01.58"

El cuadro muestra los tiempos empleados por las diferentes rutinas para ordenar matrices numéricas de tamaño variable, entre 10 y 1.000 elementos. Las pruebas se realizaron compilando el programa SortDemo (Listado 7) con AC/Basic y QuickSortDemo (Listado 8) con True Basic. Los tiempos correspondientes a la rutina QuickSort no deben compararse directamente con los demás, dado que esa rutina está escrita y compilada con True BASIC, y la velocidad básica de ejecución es diferente.

```
' BubbleSort 1.0                                                          .265

SUB BubbleSort (n(),low,high) STATIC        ' Ordenación por BURBUJA      . 86
  FOR i=low TO high-1                       ' Para cada elemento...       .258
    FOR j=low TO high-1                     ' Comprobar todos los demás   .601
      IF n(j)>n(j+1) THEN SWAP n(j),n(j+1)  ' Intercambiar                .672
    NEXT                                                                  .  6
  NEXT                                                                    .959
END SUB                                                                   .214

Numero de lineas: 8
```

LISTADO 1

```
' QuickBubble 1.0                                                         . 77

SUB QuickBubbleSort (n(),low,high) STATIC   ' Ordenación BURBUJA-rápida   .111
  i=1:Ok=0                                  ' "Ok" es el indicador        .270
  WHILE Ok=0 AND i<=high-1                  ' Si está a cero, fin         .363
    Ok=1                                                                  .886
    FOR j=low TO high-1                     ' (Igual que BubbleSort)      .996
      IF n(j)>n(j+1) THEN                                                 .922
        SWAP n(j),n(j+1)                                                  .811
        Ok=0                                ' Desactivar indicador        . 19
      END IF                                                              .513
    NEXT                                                                  .  6
    i=i+1                                                                 .227
  WEND                                                                    .987
END SUB                                                                   .214

Numero de lineas: 14
```

LISTADO 2

```
' Shell 1.0                                                               .876

SUB ShellSort(n(),low,high) STATIC          ' Ordenación SHELL            .154
  Ancho=high-low+1                          ' Ancho inicial = todo        .504
  WHILE Ancho>1                             ' Hasta que Ancho=1...        . 92
    Ancho=Ancho\2                           ' Ancho = mitad de Ancho      .437
    Flag=0                                  ' Indicador                   .241
    WHILE Flag=0                            ' Bucle                       .751
```

Al estar preparadas en forma de SUBs, estas rutinas de ordenación se pueden incluir fácilmente en cualquier programa.

adicional, llamada Swap que equivale a la función SWAP de Amiga Basic (no incluida en True BASIC) y que sirve para intercambiar dos variables. La variable Randy es la que selecciona el pivote de forma aleatoria. Partition es el valor del pivote. Si se desea utilizar siempre el punto central en vez de un punto aleatorio, basta con cambiar la línea esta:

```
LET randy=INT((high-low+1)/2)
```

La parte final de la rutina realiza la llamada recursiva a QuickSort. ¿Por qué hay cuatro llamadas en vez de dos? La razón es importante: Las llamadas recursivas emplean bastante memoria del stack (pila) del intérprete o compilador. Muchas llamadas recursivas pueden producir un error «Out of Memory». Si se llama primero a QuickSort para que ordene la parte más pequeña, esta lista se liberará antes, ahorrando un poco de memoria. Esta operación puede ser crítica en algunas ocasiones. Si la rutina pudiera adaptarse a Amiga Basic, habría que ampliar la memoria para STACK, mediante una instrucción como CLEAR ,,4096, aunque esto depende del número de elementos que se tenga previsto ordenar. (No es éste el caso de True BASIC, donde no parecen importar las necesidades de STACK, al menos hasta 2.500 elementos).

La rutina QuickSort puede ordenar de forma casi instantánea hasta 100 elementos. Los tiempos de la tabla del cuadro 3 para la rutina QuickSort son significativos pero no pueden compararse con los de las demás rutinas, al estar compiladas bajo distintos sistemas.

Otras variantes

Cualquier usuario puede emplear estas rutinas en sus propios programas. Después de encontrar la que mejor se ajuste a sus necesidades, podrá crear sus propias variantes: rutinas que ordenen de mayor a menor (ordenación inversa), que traten matrices alfanuméricas, o de varias dimensiones.

En ocasiones, para ordenar de forma inversa una matriz no es necesario cambiar la estructura ni repetir la rutina cambiando los «Menor que» por «Mayor que» y viceversa. Utilizando un pequeño truco de lógica booleana se puede adaptar cualquiera de estas rutinas. Al principio se deben inicializar las siguientes variables:

```
False = 0
True = NOT False
HaciaArriba =
```

En la variable HaciaArriba se puede colocar el valor True (verdadero, -1) si se desea orde-

```
      Flag=1                                      ' Activar indicador           .450
      FOR j=low TO high-Ancho                                                   .806
        i=j+Ancho                                 ' Calcular pareja             .782
        IF n(j)>n(i) THEN                         ' ... desordenados?           .401
          SWAP n(i),n(j):Flag=0                   ' Intercambiar por pares      .309
        END IF                                                                  .422
      NEXT                                                                      . 38
    WEND                                                                        . 34
  WEND                                                                          .987
END SUB                                                                         .214

Numero de lineas: 17
```

LISTADO 3

```
  Select 1.0
                                                                                .275
SUB SelectSort(n(),low,high) STATIC               ' Ordenación por SELECCION    .913
  FOR i=low TO high-1                             ' Comprobar todos             .425
    Max=i                                         ' Maximo = actual             .703
    FOR j=i+1 TO high                             ' Todos los demas...          .334
      IF n(j)<n(Max) THEN Max=j                   ' Si es mayor, marcar         .962
    NEXT                                                                        .  6
    SWAP n(i),n(Max)                              ' Intercambiar                .672
  NEXT                                                                          .959
END SUB                                                                         .214

Numero de lineas: 10
```

LISTADO 4

```
' Insert 1.0
                                                                                .716
SUB InsertSort(n(),low,high) STATIC               ' Ordenación por INSERCION    .607
  FOR i=low+1 TO high                             ' Para cada elemento          .220
    Aux=n(i)                                      ' Valor auxiliar=actual       .586
    j=i-1                                         ' Insertar=anterior           .136
    WHILE Aux<n(j) AND j>=low                     ' si hay que insertar...      .779
      n(j+1)=n(j)                                 ' Copiar anterior             .755
      j=j-1                                       ' Siguiente inserción         . 32
    WEND                                                                        . 34
    n(j+1)=Aux                                    ' Intercambiar                .390
  NEXT                                                                          .959
END SUB                                                                         .214

Numero de lineas: 12
```

LISTADO 5

```
! QuickSort 1.0 para True BASIC                                                 .152
! (C)1990 By Alvaro Ibáñez                                                      .881

SUB QuickSort(n(),low,high)                                                     .561
  IF low<high THEN                              ! ... Fin de ordenacion?        .995
    IF high-low=1 THEN                          ! ... Solo dos elementos?       .236
      IF n(low)>n(high) THEN                    ! Solo dos, y cambiados...      . 20
        CALL Swap(n(low),n(high))               ! ... ordenarlos                .532
      END IF                                                                    .513
    ELSE                                                                        .428
    LET randy=INT(RND*(high-low+1))+low         ! Obtener punto "pivote"        .539
    CALL Swap(n(high),n(randy))                 ! Llevarlo hasta el final       . 47
    LET partition=n(high)                       ! Partición = mayor             . 20
    DO                                                                          .683
      LET i=low                                                                 .966
      LET j=high                                ! Mover indicador               .285
      DO WHILE i<j AND n(i)<=partition          ! en ambas direcciones          .108
        LET i=i+1                               ! hacia el pivote central       .275
      LOOP                                                                      .419
      DO WHILE j>i AND n(j)>=partition                                          .112
        LET j=j-1                                                               .422
      LOOP                                                                      .419
      IF i<j THEN                               ! En el centro, intercambiar?   .873
        CALL Swap(n(i),n(j))                    ! Si                            .802
      END IF                                                                    .513
    LOOP WHILE i<j                                                              .188
    CALL Swap(n(i),n(high))                     ! Mover pivote a posicion       .424
```

```
    IF (i-low) < (high-i) THEN          ! Llamadas recursivas...        .686
      CALL QuickSort (n,low,i-1)        ! Ordenar parte mas pequeña     .457
      CALL QuickSort (n,i+1,high)       ! Ordenar parte mas grande      .214
    ELSE                                                                .428
      CALL QuickSort (n,i+1,high)       ! Ordenar parte mas pequeña     .732
      CALL QuickSort (n,low,i-1)        ! Ordenar parte mas grande      . 32
    END IF                                                              .956
  END IF                                                                .700
END IF                                                                  .654
END SUB                                                                 .214

SUB Swap(a,b)                           ! SWAP: Intercambiar A y B      .171
  LET t=a                               ! T es una variable temporal    .181
  LET a=b                                                               .242
  LET b=t                                                               .711
END SUB                                                                 .214

Numero de lineas: 41
```

LISTADO 6

```
' Demostración rutinas de ordenación - Amiga Basic                      .589
' (c)1990 by Alvaro Ibáñez                                              .429

CLS                                                                     .313
RANDOMIZE TIMER                                                         . 96
WINDOW 1,"DEMO",(50,20)-(550,180)                                       .723
WIDTH 60                                                                .902

' Función ZFill                                                         .196
' Rellena números con ceros a la izquierda                              .  5
DEF FN ZFill$(num,dig)=RIGHT$(STRING$(dig,48)+MID$(STR$(num),2),dig)    .840

PRINT "Demostración rutinas de ordenación"                              .951
PRINT "(C)1989 by Alvaro Ibáñez"                                        .321
PRINT                                                                   .463

' Introducción de datos                                                 .973
INPUT "Número de elementos [10] ";n                                     .650
IF n=0 THEN n=10                                                        .564
INPUT "Visualizar resultados para comprobación [n] ";v$                 .340
IF v$="" THEN v$="n"                                                    .342
PRINT                                                                   .463

DIM n(n),Master(n)                      ' Dimensionar matriz            .336
a=1:b=n                                 ' Elementos inicial,final       .948

FOR i=a TO b                            ' Generar matriz maestra        .446
  Master(i)=INT(RND(1)*(b-a)+1)         ' de forma aleatoria            .811
NEXT                                                                    . 61

FOR type=1 TO 5                         ' Para los cinco tipos...       .487
  PRINT "Método";type", ordenando... ";                                 .901

  FOR i=a TO b                          ' Preparar la matriz n()        .804
    n(i)=Master(i)                                                      .937
  NEXT                                                                  .959

  t=TIMER                                                               .755
  ON type GOSUB Bubble,QuickBubble,Shell,Slect,Insert                   . 56
  t=TIMER-t                                                             . 52

  IF UCASE$(v$)<>"N" THEN               ' Imprimir resultados           .141
    PRINT                                                               . 78
    FOR i=a TO b                                                        .508
      PRINT USING "### ";n(i);                                          .801
    NEXT                                                                .  6
    PRINT                                                               . 78
  END IF                                                                .700

  PRINT "Tiempo empleado: ";            ' Imprimir tiempo               .158
  Min=INT(t/60)                                                         .725
  Sec=INT(t-Min*60)                                                     .973
  Cen=t-INT(t)                                                          .723
  PRINT FN ZFill$(Min,2);":";                                           .170
  PRINT FN ZFill$(Sec,2);".";                                           .305
  PRINT FN ZFill$(Cen,2)                                                .140
  PRINT                                                                 .755
  BEEP                                                                  .159

NEXT                                                                    . 61
```

nación ascendente, o False (falso, 0) para ordenación descendente. En la rutina, se deben sustituir las expresiones como:

IF n(j) > n(i) THEN ...

por

IF (n(j) > n(i))=HaciaArriba THEN ...

El resultado de la expresión (n(j) > n(i)) es siempre 0 ó -1 (verdadero o falso). Al comparar este valor con HaciaArriba (que también es verdadero o falso), se obtienen el resultado deseado de forma automática.

La mayoría de las rutinas necesitan un solo cambio de este tipo, aunque en QuickSort, por ejemplo, serían necesarios varios cambios.

Para la ordenación de matrices alfanuméricas sólo hay que sustituir todas las apariciones de n() por n$(). Como es bien sabido, el Basic entiende que una cadena es «mayor» que otra teniendo en cuenta el orden alfabético de los caracteres que la componen.

Ordenación de ficheros en disco

Un paso más allá en las posibilidades de los sistemas de ordenación de estas rutinas lleva a la ordenación automática de ficheros, tanto secuenciales (por líneas) como directos (por registros). ¿Que sería necesario? Pueden darse dos casos:

1. Si las líneas o registros pueden introducirse en una matriz, la solución está clara: Sólo es necesaria una rutina que transfiera los datos del fichero a la memoria, para después llamar a la rutina de ordenación y finalmente volver a transferir los datos de la memoria al fichero.

2. Si el volumen de datos es demasiado grande para la memoria disponible, hay que realizar la operación directamente sobre el disco. Antes de cada comparación deben leerse del fichero los dos registros a comparar, y grabarlos después de los intercambios. Este sistema sólo es válido para los ficheros aleatorios y sorprendentemente rápido y efectivo si se emplea disco duro.

En el caso de los ficheros aleatorios, un pequeño truco permite acelerar la velocidad de acceso, de programación y de manejo de variable: definir un campo (FIELD) con el contenido de todo registro (Amiga Basic permite definir distintas variables para un mismo registro). Ejemplo:

```
OPEN «Fichero» AS 1 LEN=80                  ' Abrir Fichero
FIELD 1, 30 AS Nombre$,50 AS Direcc$        ' Definición campos
FIELD 1,100 AS Registro$                    ' Registro global
...
```

QuickSort es el método de ordenación más eficaz, y se emplea de forma estándar en muchas aplicaciones y compiladores.

```
' Rutina de ordenación
'
GET #1,i: a$=Registro$                      ' Registro 1
GET #1,j: b$=Registro$                      ' Registro 2
IF a$ > b$ THEN ...                         ' Comparación...
'
LSET Registro$=a$:PUT #1,i                  ' Guardar 1
LSET Registro$=b$:PUT #1,j                  ' Guardar 2
'
```

En el intercambio de variables, deben cambiarse a$ y b$ si es necesario. Lo mejor es utilizar variables auxiliares (a$, b$) porque las variables de FIELD no pueden asignarse directamente.

Un truco más (a veces la programación en Basic permite más trucos que la chistera de un mago) consiste en definir un campo falso inicial, de forma que se pueda ordenar el fichero por el resto de los campos, creándose así una especie de índice. Para ello bastaría añadir las siguientes líneas al principio del programa:

```
Longitud=80                                 ' Longitud Fichero
Saltar =30                                  ' Bytes a ignorar
'
FIELD 1, Saltar AS Nulo$, Longitud-Saltar AS Indice$
```

Al realizar el GET del fichero, se debe asignar Registro$ del mismo modo que antes, pero también se deben asignar otras dos variables, por ejemplo i1$=Indice$ e i2$=Indice$. La comparación se realiza entonces con la variable Indice$ en vez de a$ y b$. Se intercambian, si es necesario, a$ y b$, y finalmente se graba Registro$, del mismo modo que antes. Hay que tener cuidado para no perder ningún dato en el intercambio de variables.

Conclusiones

El cuadro 3 muestra unas pruebas de velocidad realizadas con las tres rutinas. Los tiempos están expresados en segundos. Como se puede observar, los tiempos para los primeros métodos crecen de forma exponencial abrumadora. Los métodos de selección e inserción pueden ser válidos hasta 100 elementos (siempre en versiones compiladas), y podrían resultar prácticos si no se desea demasiada complicación a la hora de programar, por ejemplo si se está adaptando alguno de estos métodos a código máquina. El método Shell es el indiscutible líder para ordenaciones no-recursivas (como las que se pueden obtener con Amiga Basic) y QuickSort para las aplicaciones más profesionales.

En definitiva, distintos métodos que demuestran cómo la programación en Basic todavía puede resultar interesante, entretenida y práctica para muchas aplicaciones, al mismo tiempo que se aprende programando. ■

```
PRINT                                                         .463
PRINT "Pulsa RETURN"                                          .233

WHILE INKEY$<>CHR$(13)                                        .369
WEND                                                          . 89

END                                                           .992

' Llamadas a las Subrutinas                                   .121
Bubble:                                                       .667
  CALL BubbleSort(n(),a,b):RETURN                             .532

QuickBubble:                                                  .210
  CALL QuickBubbleSort(n(),a,b):RETURN                        .363

Shell:                                                        .847
 CALL ShellSort(n(),a,b):RETURN                               .257

Slect:                                                        .445
  CALL SelectSort(n(),a,b):RETURN                             . 30

Insert:                                                       .882
 CALL InsertSort(n(),a,b):RETURN                              .502

' *** Insertar aquí las rutinas de ordenación ***             .659

Numero de lineas: 65
```

LISTADO 7

```
! Demo QUICKSORT 1.0                                                      . 5
! (c)1990 by Alvaro Ibañez                                                .364
! Adaptado a True Basic/Amiga partiendo de las versiones                  .987
! originales de Turbo Basic / QuickBasic 4                                .782
!                                                                         .231
! Utilizacion: CALL QuickSort (Matriz(),Primero,Ultimo)                   .126
! Resultado  : La parte indicada de Matriz() queda ordenada               .199
! *** Advertencia!!! ***                                                  .649
! Este programa solo funciona correctamente bajo True BASIC               .487
CLEAR                                                                     .308
DIM a(5000)                           ! Dimensionar matriz                . 98
PRINT "Demostracion rutina QuickSort"                                     .760
PRINT "(C)1990 by Alvaro Ibañez"                                          .857
PRINT                                                                     .463
INPUT PROMPT "Numero de elementos (1-5000)? ": Num                        .174
PRINT                                                                     .463
PRINT "Generando Matriz..."                                               .965
FOR i=1 TO num                        ! Generar matriz de numeros         .168
  LET a(i)=INT(RND*num+1)             ! Generar numero al azar            .678
  PRINT a(i);                         ! Imprimirlo                        .310
NEXT i                                                                    .297
PRINT                                                                     .463
LET t=TIME                                                                .777
CALL QuickSort(a,1,num)               ! Llamada a QuickSort               .855
LET t=TIME-t                                                              .131
PRINT "Tiempo empleado: ";            ! Imprimir tiempo                   .951
LET Min=INT(t/60)                                                         .385
LET Sec=INT(t-Min*60)                                                     .816
LET Cen=t-INT(t)                                                          .389
PRINT Min;":";                                                            .740
PRINT Sec;                                                                .222
PRINT Cen                                                                 .413
PRINT                                                                     .463
PRINT                                                                     .463
PRINT "Matriz ordenada!"                                                  .733
FOR i=1 TO num                        ! Imprimir matriz ordenada          .928
  PRINT a(i);                                                             .967
NEXT i                                                                    .297

PRINT                                                                     .463
LINE INPUT PROMPT "Pulsa RETURN": a$                                      .986
END                                                                       .992
! *** Insertar aqui la rutina QuickSort ***                               .698

Numero de lineas: 42
```

LISTADO 8

ANIM



DEL P

Este puntero animado habrá hecho pensar, probablemente a más de uno, que esto sólo se podría conseguir mediante un sofisticado programa. En este artículo se mostrará cómo conseguirlo en uno de los lenguajes más fáciles y populares: el AMIGABASIC.

Seguramente ya sabrán que su principal función es la de «apuntar» a los diferentes Iconos, Gadgets, elementos de trabajo, etc.; sin embargo mediante la animación se conseguirá un uso más elegante del mismo.

Para empezar decir que dicho puntero no está limitado a tener siempre la misma imagen, pues con el programa PREFERENCES se pueden cambiar este y otros parámetros diferentes como pueden ser el modo gráfico, opciones para impresora, etc.

Centrándonos en el Preferences y más concretamente en su opción EDIT POINTER, comprobarán que cambiar la imagen del puntero no es tan difícil, ya que sólo deberán limitarse a crear una nueva imagen con las diferentes ayudas del editor.

Una vez se ha creado la nueva imagen, se dispone de tres opciones: Cancelar, Usar y por último Salvar. Si se elige esta última opción, los diferentes valores de la imagen creada, junto con otros parámetros serán grabados en el fichero SYSTEM-CONFIGURATION (del directorio DEVS).

Pero... ¿por qué me cuenta todo esto? Muy sencillo: la base de cualquier animación, es en realidad el resultado de representar diversas imágenes en una determinada secuencia y a una determinada velocidad; con lo cual se conseguirá dar una cierta sensación de movimiento. Como les he explicado anteriormente habrán comprobado que es posible cambiar la imagen del puntero así como grabar dicha imagen en un fichero. Así se deduce que es posible animar dicho objeto con la creación de diferentes secuencias.

La estructura del Puntero

El puntero se corresponde realmente con el Sprite 0 del Amiga. Seguramente ya sabrán que la estructura de un Sprite es de 16 pixels de alto por 4 de ancho, además de poder utilizar un máximo de cuatro colores.

Sabiendo que la cantidad de colores está relacionada proporcionalmente con la cantidad de Bitplanos (planos de bit), se deduce que el total de planos es de dos (SQR (4)), por lo que en

...ACION ...PUNTERO

Las secuencias (o animaciones) creadas con los programas HUNTPOINTER o PROGANIM (así como los propios programas) sobre un modelo AMIGA, podrán ser cargadas (o ejecutados) en otros modelos diferentes al utilizado.





Habrán podido observar, seguramente todos los días, un gráfico que se mueve a través de la pantalla del Amiga a medida que se mueve el ratón. También habrán podido observar, en alguno de los discos publicados por la revista, un puntero animado.

Por F. Javier Rodríguez



Si se desea adquirir mayor velocidad de animación, se recomienda compilar el programa con cualquiera de los compiladores existentes.

```
'                                                           . 273
'       ANIMACION DE PUNTERO                                . 120
'                                                           . 273
'       (C) 1990  AMIGA WORLD                               . 288
'                                                           . 273
'       POR:  F. JAVIER RODRiGUEZ                           . 877

CLEAR 25000                                                 . 443
WINDOW CLOSE 1 ·                                            . 224
WINDOW 1,"ANIMACION PUNTERO",(0,0)-(360,80),28,-1           . 108
WINDOW OUTPUT 1                                             . 914
CHDIR "DF0:"                                                . 601


MENU OFF  ' ES ACONSEJABLE INTRODUCIR                       . 37
'           ESTA LiNEA EN LTIMO LUGAR                       . 277
'                                                           . 273
'    DECLARACIoN DE VARIABLES                               . 279
'                                                           . 273

BBASE(1)= 20212                                             . 267
BBASE(2)= 3204                                              . 144
BBASE(3)= 9956                                              . 798
DIM BYTESANIM (640)                                         . 205
DIM BUFFER (64)                                             . 341
DIM NOM$ (10)                                               . 583

ON ERROR GOTO TRATERR                                       . 938
MEMORIA:                                                    . 220

CLS :PRINT                                                  . 677
PRINT "CANTIDAD DE MEMORIA DE SU ORDENADOR"                 . 978
PRINT :PRINT "1- 512K"                                      . 326
PRINT "2- 1 MEGA"                                           . 918
PRINT "3- 2 MEGAS" :PRINT                                   . 644
LINE INPUT "INTRODUZCA OPCIoN ";MEM$                        . 477
IF VAL(MEM$) <1 OR VAL(MEM$)>3 THEN MEMORIA                 . 130
MEM = BBASE (VAL(MEM$))                                     . 114
'                                                           . 273
'    AHORA SALVA EN BUFFER ACTUAL IMaGEN DEL PUNTERO         . 841
'                                                           . 273
FOR T=MEM TO MEM+64:BUFFER (T-MEM)=PEEK (T)                 . 61
NEXT T                                                      . 150

WINDOW 1,"PROGANIM",(0,0)-(360,40),0,-1                     . 745
CLS:PRINT                                                   . 933
LINE INPUT"?ES UNA ANIMACIoN? (S/N) ";R$                    . 341
IF R$="S" OR R$="S" THEN GOSUB CARGANIM:GOTO RETARDO        . 998

NUMFICHE:                                                   . 301

CLS:PRINT                                                   . 933
LINE INPUT "CANTIDAD DE FICHEROS PARA ANIMACIoN ";FICHE$    . 50
IF VAL(FICHE$)<=0 THEN NUMFICHE                             . 518
IF VAL (FICHE$)>10 THEN PRINT:PRINT "MaXIMO DISPONIBLE 10   . 244
FICHEROS":GOSUB RATON:GOTO NUMFICHE                         . 421
FICHE= VAL (FICHE$)                                         . 196
TOTALBYTES = FICHE*64                                       . 644
'                                                           . 273
'    AHORA SALVA EN BUFFER ACTUAL IMaGEN DEL PUNTERO         . 841
'                                                           . 273
FOR T=MEM TO MEM+64:BUFFER (T-MEM)=PEEK (T)                 . 61
NEXT T                                                      . 150
'                                                           . 273
'                                                           . 273
'      AQUi SE INTRODUCE EL NOMBRE DE LOS FICHEROS          . 416
'                                                           . 273
FOR T=1 TO FICHE                                            . 666
CLS                                                         . 313
PRINT:PRINT "NOMBRE FICHERO -";T                            . 619
PRINT:LINE INPUT NOM$(T)                                    . 825
NEXT                                                        . 61

LONJ=0                                                      . 557
WHILE FICHE >0                                              . 740
OPEN NOM$(FICHE) FOR INPUT AS 1                             . 136
FICHE=FICHE-1                                               . 200
FOR T=0 TO 63                                               . 791
BYTESANIM(LONJ)=ASC((INPUT$(1,#1)))                         . 65
LONJ=LONJ+1                                                 . 746
NEXT                                                        . 61
CLOSE 1                                                     . 56
WEND                                                        . 89
```

▶

total la cantidad de bytes utilizados por el puntero será de 64 bytes ((16y X 4x) X 2 bitplanes).

Dentro de la estructura del Puntero (o Sprite), decir que los bytes se encuentran representados en el siguiente orden: primero el primer byte de la primera fila en el primer bitplano, después el primer byte de la primera fila del segundo bitplano, a continuación el segundo byte de la primera fila del primer bitplano, segundo byte de la primera fila del segundo bitplano... hasta completar las 16 filas existentes.

Comenzando a animar

Una vez conocida la principal estructura del Puntero, es imprescindible conocer en qué porción de memoria se encuentran almacenados los datos del mismo; pues bien a continuación podrán encontrar dichos valores, los cuales dependen directamente de la memoria que posea su ordenador. A continuación podrán encontrar las direcciones de los principales modelos (o modelos con ampliaciones):

Modelo	Dir. inicial	Dir. final
Amiga 512k	20212	20275
Amiga 1Mega	3204	3267
Amiga 2Megas	9956	10019

Conociendo dichos datos será posible cambiar la imagen del puntero fuera de cualquier editor. Pruebe a introducir la siguiente línea desde el AmigaBasic:

```
FOR T=DIR.INICIAL TO DIR.FINAL: POKE T,INT
        (RND(1)*255):NEXT
```

Podrá comprobar cómo la imagen del puntero cambia, de la que tenía a la resultante de 63 valores aleatorios. No es una imagen muy bonita, pero sí es el principio de la animación.

A continuación introduzca las siguientes líneas Basic:

```
Lazo:
 FOR T=DIR.INICIAL TO DIR.FINAL
 POKE T,INT (RND(1)*255): NEXT
GOTO Lazo
```

El resultado de este programa será su primera animación del puntero, aunque ésta se limíte a una cantidad de puntos en movimiento. Habrán observado que sí es posible animar el puntero desde el Basic, pero se presentan unos cuantos inconvenientes:

I. El funcionamiento del puntero animado se reduce a la ventana Basic, solución: cerrar la ventana en la primera línea Basic.



Gral. Franco, 7 - entreplanta F y G.
34001 PALENCIA.
Teléfonos (988) 75 11 60, Fax (988) 75 11 91.





GREAT VALLEY PRODUCTS

CONVIERTA HOY MISMO SU A 2000 EN:

A 2500 + 1

KIT 2501 16 MHZ

— Placa aceleradora con el Microprocesador 68030 a 16 MHZ .... 139.720 ptas.
— Coprocesador 68882/16 MHZ .... 63.000 ptas.
— Memoria RAM de 4 MEG/32 BITS.. 363.720 ptas.

KIT COMPLETO: 531.720 ptas.

A 3000 + 1

KIT 3001 28 MHZ

— Placa aceleradora con el Microprocesador 68030 a 28 MHZ .... 202.710 ptas.
— Coprocesador 68882/28 MHZ .... 101.214 ptas.
— Memoria RAM de 4 MEG/32 BITS.. 363.720 ptas.

KIT COMPLETO: 637.710 ptas.

OPCIONES PARA ESTOS KITS

— Disco duro 40 MB (11 Ms) .... 108.920 ptas.
— Disco duro 80 MB (11 Ms) .... 189.000 ptas.
— Ampliación de 4 MEG. adicionales . 146.566 ptas.

DISCOS DUROS REMOVIBLES

— Unidad lectora .... 180.614 ptas.
— Cartucho de 42 MEG. (28 Ms) ...... 25.270 ptas.
— Placa controladora con OMEG ..... 66.000 ptas.

SISTEMAS SEGURIDAD STREAMERS

— IMPACT unidad lectora WT 150 .... 159.334 ptas.
— Cinta de 150 MEG .... 33.250 ptas.
— Placa controladora 0 MEG .... 66.000 ptas.

AMPLIACIONES DE MEMORIA

— Placa ampliación de memoria 2/4/6 y 8 MG. Controlador de Disco Duro incluido. NORMA SCSI con 0 MG .... 79.900 ptas.

DISCOS DUROS

	De placa para el A 2000		Externos para A 500
	Normal	Plus (*)	
IMPACT 20 MG (40 Ms)	115.400	139.900	—
IMPACT 45 MG (40 Ms)	140.600	148.428	177.954
IMPACT 40 Q (11 Ms)	159.000	158.004	190.190
IMPACT 80 Q (11 Ms)	229.000	232.750	265.734
IMPACT 100 Q (11 Ms)	249.000	252.168	292.334

(*) La versión Plus permite añadir 2 MG de RAM en la misma placa controladora.

A ESTOS PRECIOS HAY QUE AÑADIRLES EL 12 % DE I.V.A.



***P**ara salir de la ventana de ERROR, en cualquiera de los programas, será necesario pulsar el botón izquierdo del ratón.*

```
RETARDO:                                                   . 881

CLS                                                        . 313
'                                                          . 273
'     AQUI PIDE RETARDO EN SEGUNDOS                        . 107
'                                                          . 273
PRINT                                                      . 463
LINE INPUT "RETARDO DE LA ANIMACIoN EN SEGUNDOS ";S$       . 209
SEG=1500*VAL(S$)                                           . 42

WINDOW 1,"PROGANIM",(0,0)-(120,0),0,-1                     . 597

ANIM:                                                      . 261

FOR T=0 TO 63                                              . 791
POKE MEM+T,BYTESANIM(OFFSET+T)                             . 237
NEXT                                                       . 61
FOR T=0 TO SEG:NEXT                                        . 983
OFFSET=OFFSET+64                                           . 37
IF OFFSET>=TOTALBYTES THEN OFFSET=0                        . 648
A$=INKEY$                                                  . 846
IF A$=CHR$(27) THEN SALANIM                                . 467
IF A$="ü" THEN GOSUB SALVANIM                              . 921
GOTO ANIM                                                  . 143

'     RUTINA PARA TRATAR ERRORES                           . 67
'                                                          . 273

TRATERR:                                                   . 594

WINDOW 2,"ERROR!!!",(30,30)-(260,60),0                     . 110
NERR= ERR                                                  . 964
PRINT:PRINT "SE HA PRODUCIDO UN ERROR DE"                  . 148
PRINT "CARGA, REVISE LOS DATOS"                            . 182
CLOSE 1                                                    . 56
GOSUB RATON                                                . 522
WINDOW CLOSE 2                                             . 231
RESUME NUMFICHE                                            . 662
RATON:                                                     . 665
WHILE MOUSE(0)<>1                                          . 526
WEND                                                       . 89
RETURN                                                     . 357

SALANIM:                                                   . 854
FOR T= 0 TO 63                                             . 433
POKE MEM+T,BUFFER(T)                                       . 988
NEXT T                                                     . 150
WINDOW CLOSE 1                                             . 224
END                                                        . 992

SALVANIM:                                                  . 358

WINDOW 2,"SALVAR ANIMACION",(20,20)-(260,40),0,-1          . 817
PRINT:LINE INPUT "NOMBRE FICHERO ";NO$                     . 819
NO$=NO$+".ANIM"                                            . 204
OPEN NO$ FOR OUTPUT AS 1                                   . 354
FOR T=0 TO TOTALBYTES                                      . 598
BYTE$=CHR$(BYTESANIM(T))                                   . 956
PRINT #1,BYTE$;                                            . 357
NEXT                                                       . 61
CLOSE                                                      . 533
WINDOW CLOSE 2                                             . 231
RETURN                                                     . 357

CARGANIM:                                                  . 45
LONJ=0                                                     . 557
WINDOW 2,"CARGAR ANIMACION",(20,20)-(300,40),0,-1          . 400
CLS:                                                       . 599
PRINT:LINE INPUT"NOMBRE DE LA ANIMACION ";NO$              . 312
NO$=NO$+".ANIM"                                            . 204
OPEN NO$ FOR INPUT AS 1                                    . 293
WHILE NOT EOF(1)                                           . 271
BYTESANIM(LONJ)=ASC(INPUT$(1,#1))                          . 164
LONJ=LONJ+1                                                . 746
WEND                                                       . 89
CLOSE                                                      . 533
WINDOW CLOSE 2                                             . 231
TOTALBYTES=LONJ-64                                         . 350
RETURN                                                     . 357


Numero de lineas:  141
```

II. Se carece de la posibilidad de tomar imágenes o grupos de imágenes, solución: crear dichas imágenes desde un editor compatible o crear un «cazaimágenes».

Estos problemas, así como otros derivados de la animación, son resueltos con la utilización de los dos programas, escritos en AmigaBasic, que se incluyen en este artículo:

HUNTPOINTER: «Caza» la imagen del puntero actual permitiendo su almacenamiento para un posterior uso.

PROGANIM: Completo programa de animación con el puntero.

Huntpointer. Descripción del programa

Gracias a la posibilidad multitarea del Amiga, es posible tener en ejecución, al mismo tiempo, programas escritos en diferentes lenguajes. Este es el principal motivo por el que se escribió este programa, ya que uno de los «problemas» es conseguir imágenes para la animación.

La presentación del programa, así como la elección de las diferentes opciones, se realiza a través de una ventana situada en la parte superior izquierda de la pantalla WorkBench.

El posicionamiento de dicha ventana, así como su tamaño, permite manejar libremente el ratón a través de otros iconos o programas que se encuentren en la pantalla WorkBench (más adelante sabrán su utilidad).

Las diferentes opciones que presenta dicho programa son las siguientes:

— Elección de memoria con la que se va a trabajar.

— Salir del programa. 'ESC'

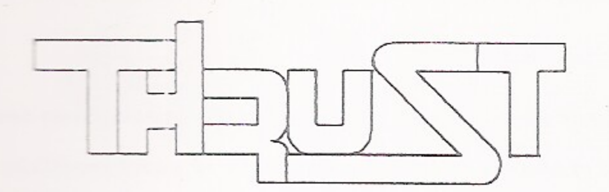
— Cazar la imagen. 'F1'

Con la primera de las opciones se deberá introducir el valor correspondiente a la cantidad de memoria que posee su Amiga. A partir de este punto el programa se limitará a un bucle continuo en el que se representarán únicamente las dos opciones restantes.

La función «Salir del programa» es evidente, pues al elegir ésta (presionando la tecla ESC) se cierra la ventana del programa; pudiendo ejecutarlo de nuevo con la opción Run de la Barra del menú.

Con la opción «Cazar imagen», el programa pedirá el nombre del fichero en el cual se almacenarán los datos (relativos a la imagen del mismo) que contiene en ese momento el puntero; a continuación se procederá a salvar los mismos.

Anteriormente se describía la posición que ocupa la ventana en la pantalla WorBench, pues bien su utilidad es la siguiente:



INFORMATIC-3

LOW COST MEMORY FOR YOUR AMIGA 500



MiniMax

... more memory for graphics and sound ...
– the variable RAM expansion from 512 K to 1.8 Megabyte –
including realtime-clock und RAM-test disk!

EXPANSIONES DE RAM

GIGATRON®

A 500 512K con Reloj..........	36.500
A 500 1MB con Reloj..........	52.500
A 1,5MB con Reloj............	68.300
A 2,0MB con Reloj............	84.300
A 500 512K no ampliable.......	25.000

¡NOVEDAD!

Nuevo Agnus para 1MB de ChipRam
Placa para Gigatron 2MB
Placa con Kickstarts conmutables
Chips 514256

A 1000 – 2000 RAM Expansión 2-8MB... llamar.

IMPRESORAS – DISKETERAS

NEC P2200....................	85.000
Epson LX 400..................	54.000
NEC P6+......................	¡llamar!
NEC P7+......................	¡llamar!
Unidades de disco 3"1/2, 5"1/4	
A 2000 int....................	20.000
NEC 1037A DRIVE con Display...	27.000
NEC 5"1/4 DRIVE con Bootselec.	33.000
NEC 5"1/4 DRIVE con Display y Bootselector.................	36.000

¡ NOVEDAD Amiga Torre !.......¡llamar!
¡ NOVEDAD Amiga 2500 !........¡llamar!

ORDENADORES – MONITORES

Monitor 1084 S(D).......	¡llamar!
A 500 KS 1.3............	¡llamar!
A 500 con 1084S.........	¡llamar!
A 2000 con 1084S........	¡llamar!
Loewe TV y Monitor......	¡llamar!

Se hacen Diversos cables para Amiga TV.

DISKETTES

Discos 3"1/2, 5"1/4	
BASF 10p. 3"1/2.........	2.550
BASF 10p. 5"1/4.........	1.000
THRUST 10p. 3"1/2.......	2.000
Nashua 10p. 3"1/2.......	2.250
Nashua 10p. 5"1/4.......	950
Joysticks:	
Competición Pro 5000....	2.900
Mouse-Pad...............	1.400

INFORMATIC 3
AVDA.DE LA ROSALEDA, 2, BAJO
TELEF.(952)22-11-28
29008 MALAGA

Servicio Técnico en Commodore, Spectrum y PC (Compatibles) ¡Llame a nuestro Servicio Técnico!
MALAGA COMPUTER Telf.(952)28-86-43



Todos los ficheros, tanto de lectura como escritura, se leerán/escribirán desde el drive df0:. Sin embargo el cambio de DRIVE o directorio se podrá modificar fácilmente.

I. Molestar mínimamente a las operaciones que se realicen en la pantalla, tales como apertura de discos, apertura de cajones, pedir INFO, etcétera.

II. Poder usar el editor de puntero EDIT POINTER para la creación de imágenes, pudiendo almacenarlas posteriormente con el programa HUNTPOINTER.

Huntpointer + Edit Pointer

Para poder crear las imágenes (con EDIT-POINTER), y posteriormente salvarlas, serán necesarios los siguientes pasos:

I. Ejecutar el programa PREFERENCES.
II. ISeleccionar EDITPOINTER.
III. Crear la imagen deseada y seleccionar OK.
IV. Seleccionar USE.
V. Activar ventana del HUNTPOINTER.
VI. Pulsar 'F1' (opción grabar imagen).
VII. Repetir estos pasos en cada imagen que se desee crear.

Este programa también resulta útil si se desea «cazar» la imagen de los punteros correspondientes a otros programas.

ProgAnim. Descripción del programa

ProgAnim es un completo programa para la animación del puntero cuyo manejo es muy sencillo. Todas las funciones así como mensajes al usuario se realizarán a través de la ventana principal, así como de la ventana de error.

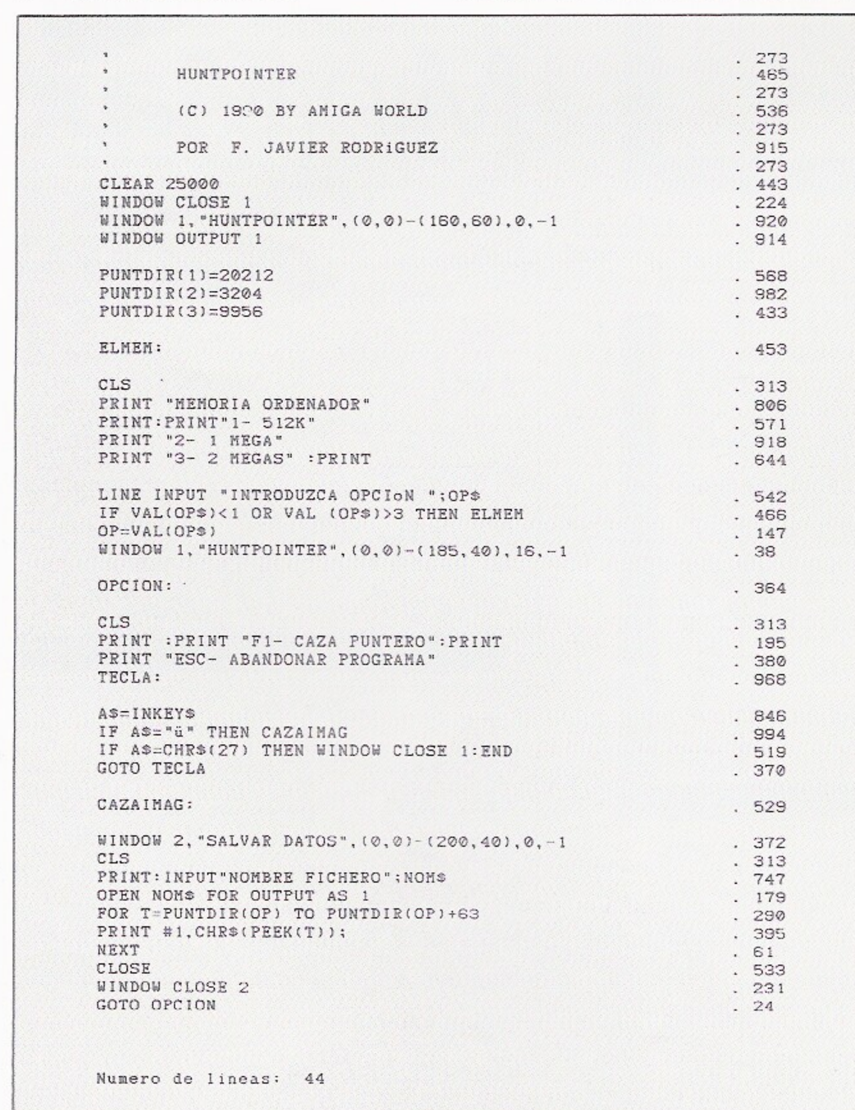
Este programa mostrará las opciones en el siguiente orden:

I. Elección de memoria con la que se va a trabajar.
II. Cargar Animación (opcional).
III. Número de secuencias para animación.
IV. Nombre de la/las diferente/s secuencia/s.
V. Retardo de la animación (en segundos).

Una vez se esté produciendo la animación se tendrá acceso a dos opciones más:

I.b. Salvar animación (todas las secuencias de las que ésta consta).

II.b. Salir del programa.



```
'                                                       . 273
'       HUNTPOINTER                                     . 465
'                                                       . 273
'       (C) 19?0 BY AMIGA WORLD                         . 536
'                                                       . 273
'       POR  F. JAVIER RODRiGUEZ                        . 915
'                                                       . 273
CLEAR 25000                                             . 443
WINDOW CLOSE 1                                          . 224
WINDOW 1,"HUNTPOINTER",(0,0)-(160,60),0,-1              . 920
WINDOW OUTPUT 1                                         . 914

PUNTDIR(1)=20212                                        . 568
PUNTDIR(2)=3204                                         . 982
PUNTDIR(3)=9956                                         . 433

ELMEM:                                                  . 453

CLS                                                     . 313
PRINT "MEMORIA ORDENADOR"                               . 806
PRINT:PRINT"1- 512K"                                    . 571
PRINT "2- 1 MEGA"                                       . 918
PRINT "3- 2 MEGAS" :PRINT                               . 644

LINE INPUT "INTRODUZCA OPCIoN ";OP$                     . 542
IF VAL(OP$)<1 OR VAL (OP$)>3 THEN ELMEM                 . 466
OP=VAL(OP$)                                             . 147
WINDOW 1,"HUNTPOINTER",(0,0)-(185,40),16,-1             . 38

OPCION:                                                 . 364

CLS                                                     . 313
PRINT :PRINT "F1- CAZA PUNTERO":PRINT                   . 195
PRINT "ESC- ABANDONAR PROGRAMA"                         . 380
TECLA:                                                  . 968

A$=INKEY$                                               . 846
IF A$="ü" THEN CAZAIMAG                                 . 994
IF A$=CHR$(27) THEN WINDOW CLOSE 1:END                  . 519
GOTO TECLA                                              . 370

CAZAIMAG:                                               . 529

WINDOW 2,"SALVAR DATOS",(0,0)-(200,40),0,-1             . 372
CLS                                                     . 313
PRINT:INPUT"NOMBRE FICHERO";NOM$                        . 747
OPEN NOM$ FOR OUTPUT AS 1                               . 179
FOR T=PUNTDIR(OP) TO PUNTDIR(OP)+63                     . 290
PRINT #1,CHR$(PEEK(T));                                 . 395
NEXT                                                    . 61
CLOSE                                                   . 533
WINDOW CLOSE 2                                          . 231
GOTO OPCION                                             . 24


Numero de lineas:  44
```

ProgAnim. Su funcionamiento

En este apartado se describirán las opciones anteriormente señaladas.

I. Se deberá seleccionar el dígito correspondiente a la memoria del Amiga.

II. Esta elección es opcional, con ella se podrá cargar una animación completa grabada anteriormente con la opción I.b.

III. Se deberá introducir el número de secuencias de las que se va a componer la animación; siendo el máximo de diez, este valor se puede cambiar con algunas variaciones en el listado. Dichos ficheros pueden corresponder a los creados con el programa HUNTPOINTER, PUNTERO MAKER (publicado en este mismo número), u otro similar.

IV. Se deberán introducir los nombres de los diferentes ficheros. Si posteriormente, durante el proceso de lectura, se produce algún error se deberá a que el fichero no es del formato adecuado o que alguno de los nombres es erróneo, debiendo introducir los datos correspondientes al número de ficheros así como los nombres de los mismos.

V. Pausa, en segundos, que se producirá entre el cambio de secuencias (valor mínimo=0).

5 I.b. Esta opción sirve para salvar a disco una animación completa, para ello debe pulsarse la tecla 'F1'. Posteriormente se podrán diferenciar las secuencias de las animaciones debido a que estas últimas se les añade el sufijo «.ANIM».

I.b. Si se desea salir de la animación se deberá pulsar la tecla 'ESC', recuperando el puntero la imagen inicial antes de ser animado (ésta se había salvado en una matriz; variable BUFFER). Asimismo se cerrará, por completo, la ventana del programa.

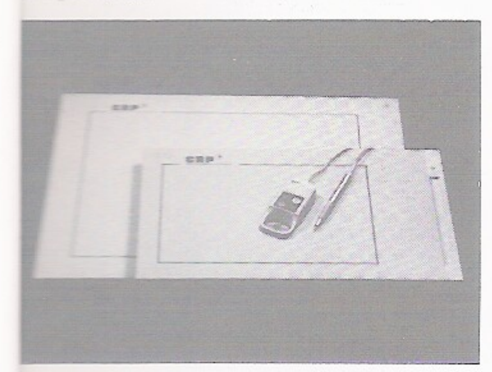
Para finalizar decirles que los programas presentados son de fácil acomodación para sus propias necesidades. ■

µ MADCOMPUTER



...ABLETAS DIGITALIZADORAS

...as tabletas digitalizadoras CRP sustitu-...en con ventaja al ratón en la creación de ...rte y diseño por computador. Trabajan ...egún el principio electrostático, y ofrecen ...na resolución de 0,1 mm (0,004") con una ...uperficie activa diseñada para cumplir las ...specificaciones DIN-A4 y DIN-A3, según ...odelo. Se encuentran disponibles para ...MIGA, ATARI, IBM-PC/XT/AT, PS/2 o ...ompatibles.



...PLICACIONES

...rquitectura ● CAD/CAM ● Construcción, ...esarrollo ● Diseño (moda, arte) ● Dibujo ...mano alzada ● Geología, Física ● Pro-...eso de imagen ● Enseñanza ● Marketing ...Emulación de teclado inteligente ● Dise-...o de interiores ● Investigación médica y ...uímica ● Técnica de menús sin teclado ● ...plicaciones musicales ● Diseño de cir-...uitos impresos ● Procesado de curvas y ...iagramas ● Transmisión de letra manus-...rita vía modem (para transmitir firmas ...nstantáneamente).

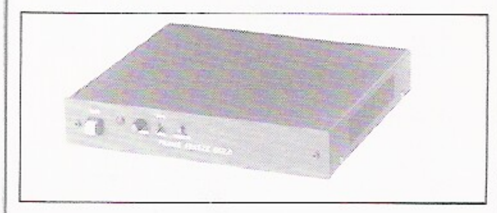
ESPECIFICACIONES

—Resolución: 0,1 mm. Emula 0,025 mm.
—Precisión: +/— 0,5 mm.
—Sup. activa: 300 mm x 210 mm (DIN-A4). 420 mm x 330 mm (DIN-A3).
—Máx. dist. lápiz/sup.: 3 mm.
—Máx. dist. para garant. prec.: 1 mm.
—Máx. vel. de muestreo: 75 pps (mod. DIN-A4). 100 pps. defin. por *software* (mod. DIN-A3).
—Emulaciones: BitPadOne y serie MM (de Summagraphics).
—Formato de datos: ASCII o binario.
—Conexión al ordenador: Puerto serie (RS 232C).
—Veloc. transm. datos: 1.200, 2.400, 4.800 o 9.600 Bd.
—Sist. de coord.: Absolutas o relativas (DIN-A3). En ambos casos, unidades métricas o pulgadas.
—Modos de func.: Punto, continuo (stream), emulación de ratón, continuo con interrupción y remoto (DIN-A3).
—Programabilidad: Sólo modelo DIN-A3.

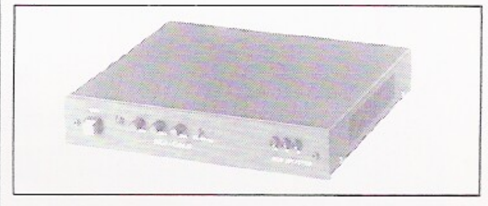


VIDEO E IMAGEN

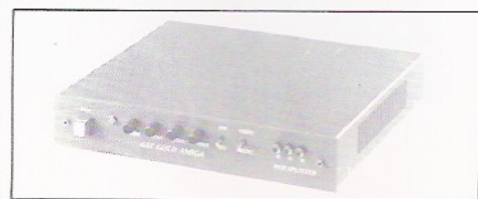
FRAME-BUFFER (congelador de imagen)
El FB GOLD permite memorizar en tiempo real (1/50 s), para su posterior digitalización, una imagen procedente de cualquier fuente de video color PAL, SECAM, NTSC, o b/n, gracias a sus 2 MB de memoria de cuadro.
Puede utilizarse con un televisor provisto de un euroconector o conectores RCA para entrada y salida de vídeo. Permite una parada perfecta de imagen de un VCR.



Filtros RGB electrónicos profesionales
Para uso con digitalizadores de vídeo en color y b/n, con resultados óptimos, sin interferencias. Ofrecen una resolución de 625 líneas entrelazadas.
El DIGI-GOLD hace la separación de componentes RGB de una señal de vídeo color (PAL o SECAM), y proporciona una señal RGB conmutada por pulsadores y sincronizada para entrada al digitalizador.
Puede usarse también para visualizar las imágenes de un VTR en un monitor RGB.
El DIGI-GOLD PRO acepta señales de entrada SECAM/PAL/Y-C (Luminancia-Crominancia) y ofrece una sección completa de KEY para luminancia y crominancia.



GENLOCKS Profesionales
El nuevo GST GOLD es un equipo profesional para la incrustación de imágenes de ordenador en video. Ofrece una resolución de 625 líneas entrelazadas y la señal de salida es conforme a la calidad BROADCAST.
Decodifica la señal de vídeo compuesto PAL o SECAM de referencia en señales RGB. Realiza la codificación de las componentes rojo, verde, azul más sincronismos en una señal de vídeo compuesto color PAL, permitiendo su grabación en un magnetoscopio, camascopio...
—Correcciones de brillo, color y contraste de la imagen de video.
—Ajuste de la fase horizontal de la imagen del ordenador.
—Incrustación sobre el blanco o el negro (detección sobre el registro cero, directa o inversa).
—Entradas y salidas PAL y RGB.
—Posee un FILTRO RGB ELECTRONICO incorporado.
—Entrada y salida de sonido (mono).
—Hay dos versiones, AMIGA y ATARI (1040 STE o serie Mega ST), disponible cada una en dos modelos: PAL y Luminancia-Crominancia (Y-C).
El GST GOLD PRO es un genlock profesional BROADCAST sin corrección de la señal de vídeo (VIDEO IN = VIDEO OUT), permite hacer CHROMA KEY sobre cualquiera de los 4.096 colores de la paleta AMIGA. Incluye FADER.



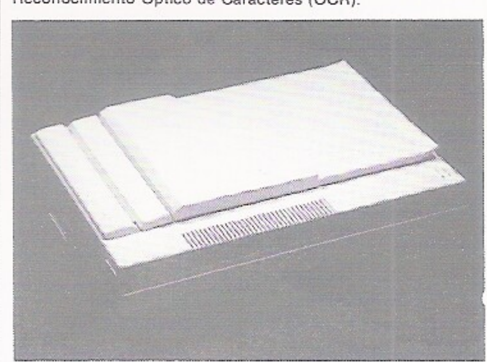
PRINT TECHNIK

SCANNERS PARA AUTOEDICION

UNIVERSAL SCANNER DIN-A4/200 dpi
El USC de PRINT-TECHNIK es un scanner de tablero con una resolución de 200 dpi y 16 niveles de gris, capaz de explorar un documento DIN-A4 en 10 segundos y que a todo ello une la posibilidad de ser utilizado como fotocopiadora e impresora térmica. Puede utilizar también otras impresoras, matricilaes e incluso laser. En sus versiones para ATARI ST/Mega ST y Commodore AMIGA, soportan todos los modos gráficos y formatos de archivo. Además, existe un programa de reconocimiento óptico de caracteres (OCR) con aprendizaje a disposición de los usuarios de ATARI. Teniendo las características estándar del Universal Scanner, la versión para IBM-PC/XT/AT y compatibles funciona con las tarjetas gráficas Hércules, CGA y EGA, pudiendo cargar/salvar imágenes en los formatos PCX, IMG y TIFF. Requiere un mínimo de 640MB de RAM.



PROFESSIONAL SCANNER II 300/600 dpi
El PSC II de PRINT-TECHNIK es un scanner de tablero, con resoluciones de hasta 300 dpi en horizontal y 600 dpi en vertical, seleccionables de modo independiente. Con 64 niveles de gris, y brillo y contraste ajustables en 8 pasos, trabaja en los modos LINE ART (Línea), HALFTONE (medios tonos) y MIXTO, lo que le permite ofrecer una salida de imágenes de gran calidad. En su versión para el Commodore AMIGA alcanza una máxima resolución de 600 x 600 dpi.
Con las características generales del PSC II, la versión para IBM-PC-XT/AT y compatibles funcona bajo MS-DOS o WINDOWS. Puede cargar/salvar imágenes en los formatos MSP (Windows Paint), PCX (PC Paintbrush), TIF (TIFF file) y TIF comprimido. El *software* que acompaña a esta versión permite un completo control del *scanner* y la manipulación de las imágenes.
Incluido en el precio del *scanner*, se ofrece un programa de Reconocimiento Optico de Caracteres (OCR).

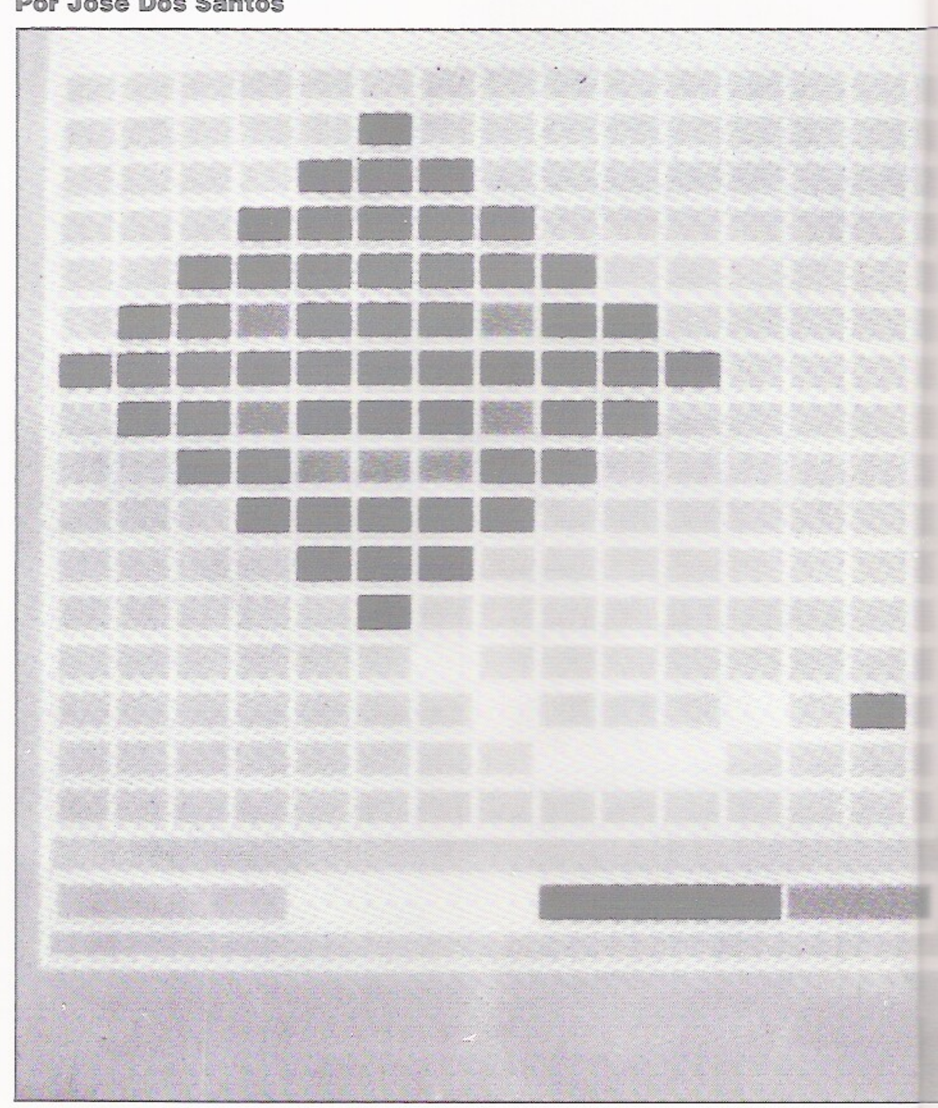


µ MADCOMPUTER
Informática Profesional
91 - 250 90 40
Suministros a toda España
Nicaragua, 4-Bajo - 28016 Madrid

Con todos los datos, referentes a la estructura del puntero, ha sido realizado el programa «Puntero Maker», el cual puede ser una gran ayuda a la hora de crear el puntero.

PUNTERO M

Por José Dos Santos



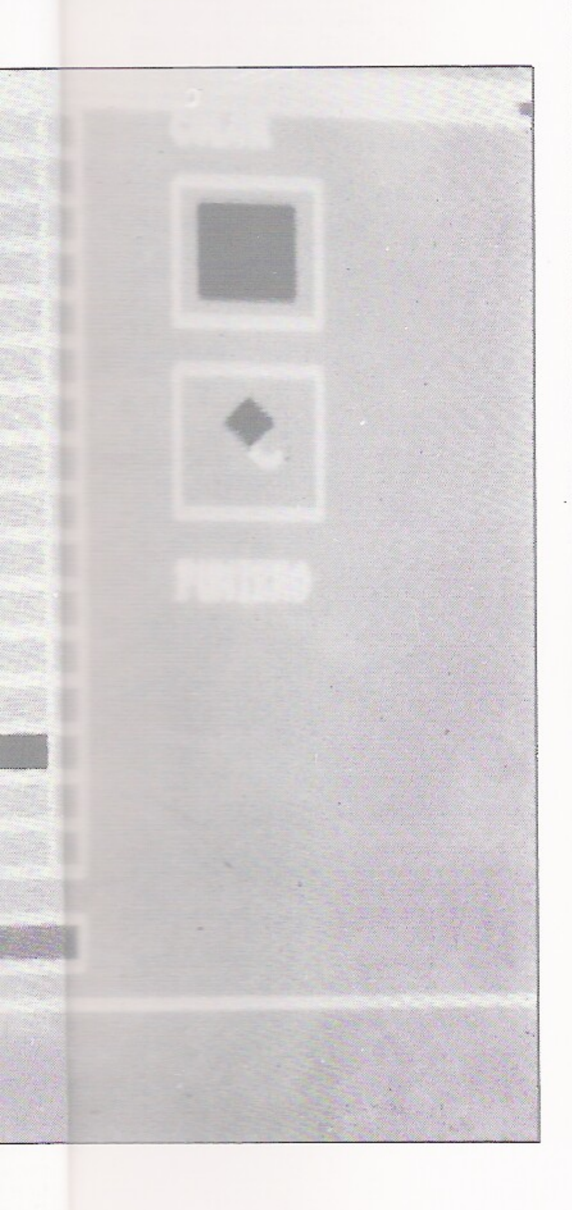
Nada más cargar el puntero en memoria se borrará la pantalla, y sobre ésta aparecerá una cuadrícula de 16x16 cuadrados, los cuales representan al puntero. Debajo de la cuadrícula habrá cuatro cuadrados que contendrán los respectivos colores del puntero, aunque en esta ocasión los colores que contiene son los de la pantalla.

A la derecha de la cuadrícula habrá dos cuadrados, uno de ellos informará al usuario del color que en esos momentos está en uso, y el cuadrado inferior contendrá la imagen del puntero tal como quedará a escala real.

Para definir el puntero basta con llevar, con el ratón, el puntero hasta la cuadrícula, colocándolo sobre el pixel que se quiera dibujar para a continuación pulsar el botón izquierdo del ratón y dibujar sobre la cuadrícula un cuadrado con el color que en ese momento se haya elegido. Para elegir el color hay que realizar el mismo proceso pero con la cuadrícula de los colores. El conjunto de todos los cuadrados de la cuadrícula que haya coloreados es lo que va a definir la forma del puntero.

Además el usuario dispone de otras ayudas que se encuentran en la barra de menús, a la cual se accede dejando pulsado el botón derecho del ratón.

MAKER



```
DIM COL(16,16):COLO=1:DIM A1$(8):DIM A2$(8):DIM INV(16,16). 186
DIM INIT(63):GOSUB CUADRICULA:PX=255:PY=0:DIM FIL$(232):   . 292
DIM VA(64)                                                 . 321
FOR N=0 TO 63:INIT(N)=PEEK(3204+N):NEXT N                  . 13
GOSUB OPCIONES:                                            . 758
ON MOUSE GOSUB PINTA:MOUSE ON:MENU ON                      . 293
ON ERROR GOTO FALLO                                        . 702
MAIN:                                                      . 951
GOTO MAIN                                                  . 527
FALLO:                                                     . 256
WINDOW 2,"ERROR",(30,30)-(260,60),15                       . 902
GOSUB PAUSA1                                               . 606
WINDOW CLOSE 2                                             . 231
MENU ON:MOUSE ON                                           . 37
RESUME MAIN                                                . 235
CUADRICULA:                                                . 86
LINE(0,0)-(400,0):LINE (400,0)-(400,160):LINE(400,160)-(0,. 77
160 )                                                      . 488
LINE(0,160)-(0,0)                                          . 148
FOR N=1 TO 16:LINE(N*25,0)-(N*25,160):NEXT N               . 213
FOR N=1 TO 16:LINE(0,N*10)-(400,N*10):NEXT N               . 138
LINE(0,170)-(400,170):LINE(400,170)-(400,180):LINE(400,180. 401
)-(0,180)                                                  . 145
LINE(0,180)-(0,170):FOR N=1 TO 4:LINE(N*100,170)-(N*100,18. 370
0 )                                                        . 337
PAINT (N*100-20,175),N-1,1:NEXT N                          . 452
LINE(440,15)-(500,15):LINE(500,15)-(500,45):LINE(500,45)-(. 893
440 ,45)                                                   . 482
LINE(440,45)-(440,15):LOCATE 1,56:PRINT"COLOR":LINE (450,2. 165
0 )-(490,40),,BF                                           . 250
LINE(440,55)-(500,55):LINE (500,55)-(500,85):LINE(500,85)-. 866
(440,85)                                                   . 245
LINE(440,85)-(440,55):LOCATE 13,56:PRINT "PUNTERO"         . 210
RETURN:                                                    . 907
PINTA:                                                     . 381
IF MOUSE(0)>=0 THEN PINTA                                  . 529
X=MOUSE(1):Y=MOUSE(2)                                      . 27
IF Y>159 THEN ELIGECOL                                     . 922
IF X>399 THEN MOUSE ON:RETURN                              . 99
Y=INT(Y/10):X=INT(X/25)                                    . 590
LINE(X*25,Y*10)-(X*25+25,Y*10+10),COLO,BF                  . 305
LINE(X*25,Y*10)-(X*25+25,Y*10):LINE(X*25+25,Y*10)-(X*25+25. 935
,Y*10+10)                                                  . 270
LINE(X*25+25,Y*10+10)-(X*25,Y*10+10):LINE(X*25,Y*10+10)-(X. 195
*25,Y*10)                                                  . 557
PSET (460+(X*2),60+Y),COLO:PSET (461+(X*2),60+Y),COLO      . 94
COL(Y,X)=COLO                                              . 850
MOUSE ON:RETURN                                            . 844
ELIGECOL:                                                  . 896
IF Y<171 OR Y>179 THEN MOUSE ON:RETURN                     . 518
IF X>399 THEN MOUSE ON:RETURN                              . 99
COLO=INT(X/100):LINE(450,20)-(490,40),COLO,BF              . 933
MOUSE ON:RETURN                                            . 844
PUNTERO:                                                   . 231
PO=0:FOR N=0 TO 31:FOR F=1 TO 8:AL=INT(N/2)                . 226
IF COL(AL,((N-AL*2)*8)+F-1)=0 THEN A1$(F)="0":A2$(F)="0"   . 46
IF COL(AL,((N-AL*2)*8)+F-1)=1 THEN A1$(F)="1":A2$(F)="0"   . 456
IF COL(AL,((N-AL*2)*8)+F-1)=2 THEN A1$(F)="0":A2$(F)="1"   . 109
IF COL(AL,((N-AL*2)*8)+F-1)=3 THEN A1$(F)="1":A2$(F)="1"   . 519
NEXT F:GOSUB PEE1:POKE 3204+AL*4+(N-AL*2),POK:GOSUB PEE2:P. 978
OKE 3206+AL*4+(N-AL*2),POK                                 . 495
PO=PO+1                                                    . 66
NEXT N                                                     . 108
PX=255-PX:PY=256-PY:IF PY=256 THEN PY=0                    . 569
POKE 12598650&,PX:POKE 12598651&,PY                        . 324
RETURN                                                     . 357
PRUBA:                                                     . 439
FOR N=0 TO 15:FOR F=0 TO 15                                . 726
PRINT COL(N,F);:NEXT F:PRINT:NEXT N                        . 316
RETURN                                                     . 357
PEE1:                                                      . 241
POK=0:BYTE=128:FOR Z=1 TO 8:IF A1$(Z)="1"THEN POK=POK+BYTE. 547
BYTE=BYTE/2:NEXT Z                                         . 865
RETURN                                                     . 357
PEE2:                                                      . 290
POK=0:BYTE=128:FOR Z=1 TO 8:IF A2$(Z)="1"THEN POK=POK+BYTE. 119
BYTE=BYTE/2:NEXT Z                                         . 865
RETURN                                                     . 357
INVERSION:                                                 . 150
FOR I=0 TO 15:FOR IN=0 TO 15:INV(I,IN)=COL(15-IN,I):NEXT I. 160
N:NEXT I                                                   . 962
FOR I=0 TO 15:FOR IN=0 TO 15:LINE(I*25,IN*10)-(I*25+25,IN*. 264
10 +10),INV(I,IN),BF                                       . 709
PSET (460+I*2,60+IN),INV(I,IN):PSET (461+I*2,60+IN),INV(I,. 686
IN)                                                        . 177
NEXT IN:NEXT I:GOSUB CUADRICULA                            . 472
FOR I=0 TO 15:FOR IN=0 TO 15:COL(I,IN)=INV(IN,I):NEXT IN:  . 899
NEXT I                                                     . 73
RETURN                                                     . 357
LOD:                                                       . 524
FOR I=0 TO 15:FOR IN=0 TO 1:FOR IL=0 TO 7:ELE(IL)=COL(I,IN. 303
*8+IL):                                                    . 752
COL(I,IN*8+IL)=COL(I,IL):COL(I,IL)=ELE(IL)                 . 805
NEXT IL:NEXT IN:NEXT I:FOR IL=0 TO 7:ELE(IL)=COL(0,8+IL):  . 757
NEXT IL                                                    . 44
FOR I=1 TO 15:FOR IN=0 TO 7:COL(I-1,8+IN)=COL(I,8+IN):     . 417
NEXT IN:NEXT I                                             . 687
FOR I=0 TO 15:FOR IN=0 TO 15:INV(I,IN)=COL(IN,I):NEXT IN:  . 548
NEXT I                                                     . 73
```

```
FOR I=0 TO 15:FOR IN=0 TO 15:LINE(I*25,IN*10)-(I*25+25,IN*. 264
10 +10),INV(I,IN),BF                                        . 709
PSET (460+I*2,60+IN),INV(I,IN):PSET (461+I*2,60+IN),INV(I,. 686
IN)                                                         . 177
NEXT IN:NEXT I:GOSUB CUADRICULA                             . 472
RETURN                                                      . 357
INVER:                                                      . 44
FOR I=0 TO 15:FOR IN=0 TO 15:INV(I,IN)=COL(IN,15-I):        . 222
NEXT IN:NEXT I                                              . 687
FOR I=0 TO 15:FOR IN=0 TO 15:LINE(I*25,IN*10)-(I*25+25,IN*. 264
10 +10),INV(I,IN),BF                                        . 709
PSET (460+I*2,60+IN),INV(I,IN):PSET (461+I*2,60+IN),INV(I,. 686
IN)                                                         . 177
NEXT IN:NEXT I:GOSUB CUADRICULA                             . 472
FOR I=0 TO 15:FOR IN=0 TO 15:COL(I,IN)=INV(IN,I):NEXT IN: . 899
NEXT I                                                      . 73
RETURN                                                      . 357
RESTAURA:                                                   . 7
FOR RES=0 TO 63:POKE(3204+RES),INIT(RES):NEXT RES:RETURN  . 550
PUNTO:                                                      . 953
MOUSE OFF                                                   . 899
IF MOUSE(0)>=0 THEN PUNTO                                   . 468
X=MOUSE(1):Y=MOUSE(2)                                       . 27
IF Y>159 THEN PUNTO                                         . 934
IF X>399 THEN PUNTO                                         . 828
Y=INT(Y/10):X=INT(X/25)                                     . 590
LINE(X*25,Y*10)-(X*25+25,Y*10+10),COLO                      . 380
LINE(X*25+25,Y*10)-(X*25,Y*10+10),COLO                      . 965
LINE(X*25,Y*10)-(X*25+25,Y*10):LINE(X*25+25,Y*10)-(X*25+25. 935
,Y*10+10)                                                   . 270
LINE(X*25+25,Y*10+10)-(X*25,Y*10+10):LINE(X*25,Y*10+10)-(X. 195
*25,Y*10)                                                   . 557
PSET (460+(X*2),60+Y),COLO:PSET (461+(X*2),60+Y),COLO     . 94
PY=Y:PX=X                                                   . 478
MOUSE ON:RETURN                                             . 844
CARGA:                                                      . 972
OPEN "DF0:DEVS/SYSTEM-CONFIGURATION" FOR INPUT AS 1:LONJ=0. 746
WHILE NOT EOF(1)                                            . 271
FIL$(LONJ)=(INPUT$(1,1))                                    . 682
LONJ=LONJ+1                                                 . 746
WEND                                                        . 89
CLOSE                                                       . 533
RETURN                                                      . 357
SAVFILE:                                                    . 262
OPEN A$ FOR OUTPUT AS 1                                     . 223
FOR GRA=0 TO 63                                             . 43
PRINT #1,CHR$(PEEK(3204+GRA));:NEXT GRA                     . 503
CLOSE                                                       . 533
RETURN                                                      . 357
LOAFILE:                                                    . 302
OPEN A$ FOR INPUT AS 1                                      . 417
FOR LONJ=0 TO 63:FIL$(31+LONJ)=(INPUT$(1,1))                . 163
NEXT LONJ                                                   . 742
CLOSE                                                       . 533
GOSUB INICIALIZA                                            . 597
RETURN                                                      . 357
GRABA:                                                      . 470
OPEN "DF0:DEVS/SYSTEM-CONFIGURATION" FOR OUTPUT AS 1        . 786
FOR GRA=0 TO 63:FIL$(32+GRA)=CHR$(PEEK(3204+GRA)):NEXT GRA. 282
FIL$(100)=CHR$(PX)                                          . 706
FIL$(101)=CHR$(PY)                                          . 804
FOR FINAL=0 TO LONJ-1:PRINT #1,FIL$(FINAL);:NEXT FINAL      . 901
CLOSE                                                       . 533
RETURN                                                      . 357
OPCIONES:                                                   . 108
MENU 1,0,1,"SISTEMA"                                        . 513
MENU 1,1,1,"GRABAR DATOS"                                   . 531
MENU 1,2,1,"CARGAR DATOS"                                   . 950
MENU 1,3,1,"GRABAR SYSTEM"                                  . 381
MENU 1,4,1,"CARGAR SYSTEM"                                  . 317
MENU 1,5,1,"SALIR         "                                 . 268
MENU 2,0,1,"EDICION"                                        . 706
MENU 2,1,1,"BORRAR"                                         . 728
MENU 2,2,1,"RESTAURAR"                                      . 541
MENU 2,3,1,"ROTAR VERT"                                     . 575
MENU 2,4,1,"ROTAR HORIZ"                                    . 312
MENU 3,0,1,"DEFINIR"                                        . 65
MENU 3,1,1,"PUNTERO"                                        . 713
MENU 3,2,1,"CURSOR"                                         . 74
MENU 4,0,1,""                                               . 924
ON MENU GOSUB ELECCION                                      . 669
RETURN                                                      . 357
SISTEMA:                                                    . 967
ON MENY GOSUB GFILE,LFILE,GSYSTEM,LSYSTEM,SALIR             . 666
RETURN                                                      . 357
SALIR:                                                      . 341
SYSTEM                                                      . 209
GFILE:                                                      . 441
P$="GRABAR DATOS"                                           . 877
FAENA$="PREGUNTA"                                           . 361
GOSUB VENTANA                                               . 32
GOSUB SAVFILE                                               . 482
RETURN                                                      . 357
LFILE:                                                      . 275
P$="CARGAR DATOS"                                           . 296
FAENA$="PREGUNTA"                                           . 361
GOSUB VENTANA                                               . 32
GOSUB LOAFILE                                               . 345
RETURN                                                      . 357
GSYSTEM:                                                    . 452
P$="GRABAR SYSTEM-CONFIGURATION"                            . 326
FAENA$="PAUSA"                                              . 356
```

▶

La barra de menús está compuesta por tres menús: el menú sistema, menú edición y menú definir.

Menú sistema

En el menú sistema, el usuario tendrá las siguientes ayudas:

— Grabar datos: sirve para grabar los bytes que componen el puntero, para después poderlos usar desde este u otro programa. El ordenador antes de realizar la operación pedirá el nombre que se quiere destinar al fichero.

— Cargar datos: sirve para cargar un fichero que se haya grabado con la anterior opción. El ordenador pide al usuario el nombre del fichero, y si éste se encuentra en el disco se procederá a su carga, dibujando sobre la cuadrícula la imagen que éste contenga.

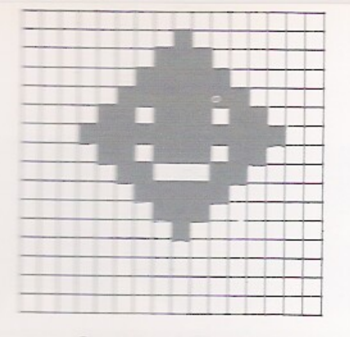
— Grabar SYSTEM: esta opción es quizás la que tenga una mayor utilidad, sirve para grabar en el fichero SYSTEM-CONFIGURATION la secuencia de bytes que componen el puntero. Cuando se elija esta opción, el ordenador pedirá que se introduzca el disco donde se quiera grabar y a continuación se pulse una tecla. En el fichero destino tiene que haberse grabado anteriormente el fichero SYSTEM-CONFIGURATION, ya que el programa no crea este fichero sino que lo único que hace es modificar los datos del mismo. Esta es la única opción que graba el bit de elección.

— Cargar SYSTEM: esta función es la inversa de la anterior. El ordenador pide que se introduzca el disco donde figura este fichero, para a continuación mostrar la imagen que contenga en la cuadrícula.

— Salir: como su propio nombre indica, esta opción cierra las opciones del Basic, saliendo al WorkBench.

Menú edición

En el menú edición el usuario podrá acceder a las siguientes opciones:



— Restaurar: si después de definir el puntero, no se ha quedado satisfecho con el nuevo aspecto que adquiere éste, eligiendo esta opción el ordenador restablece la forma original del puntero que tenía antes de haber cargado el programa.

— Rotar Vert: sirve para girar verticalmente la figura que esté en la cuadrícula 180 grados.

— Rotar Horiz: realiza la misma función que la opción anterior, sólo que en sentido horizontal.

Menú definir

Este último menú sólo está constituido por dos opciones:

— Puntero: transfiere el dibujo de la cuadrícula al puntero. Cuando se ha elegido esta opción, es recomendable no mover el ratón mientras se está definiendo.

— Cursor: sirve para definir el bit de elección del puntero. Una vez se haya accedido a esta opción, el siguiente bit que se marque en la cuadrícula será el bit de elección. Este estará representado por una "X" en el color escogido.

Cuando se elige en el menú de sistema alguna opción de grabación, lo que se graba no es el dibujo que esté en esos momentos en la cuadrícula, sino los datos que contiene el puntero.

A este artículo acompaña un pequeño programa que sirve de demostración de los curiosos efectos que se pueden realizar con el puntero.

Después de cargar el programa de demostración, éste cargará a su vez cuatro ficheros de datos que han sido creados con el PUNTERO-MAKER. Estos ficheros son el dibujo de la flecha, apuntando a cada una de las cuatro direcciones.

Cuando ya esté en marcha el programa, el usuario puede mover el puntero con toda facilidad, pero cada vez que pulse el botón izquierdo del ratón, éste cambiará de forma apuntando la flecha al cuadrante en el que se encuentre.

Sirva también como ejemplo de los curiosos efectos de animación con el puntero, el programa puntero que acompaña a los discos de AMIGA WORLD. ■

```
GOSUB VENTANA                                                   . 32
GOSUB CARGA                                                     . 994
GOSUB GRABA                                                     . 66
RETURN                                                          . 357
LSYSTEM:                                                        . 310
P$="CARGAR SYSTEM-CONFIGURATION"                                . 834
FAENA$="PAUSA"                                                  . 356
GOSUB VENTANA                                                   . 32
GOSUB CARGA                                                     . 994
GOSUB INICIALIZA                                                . 597
RETURN                                                          . 357
VENTANA:                                                        . 109
WINDOW 2,P$,(30,30)-(400,60),15                                 . 315
WINDOW OUTPUT 2                                                 . 921
IF FAENA$="PAUSA" THEN GOSUB PAUSA                              . 309
IF FAENA$="PREGUNTA" THEN GOSUB PREGUNTA                        . 34
WINDOW CLOSE 2                                                  . 231
RETURN                                                          . 357
PAUSA:                                                          . 271
PRINT "INTRODUCE EL DISCO DESTINO Y PULSA UNA TECLA"            . 831
PAUSA1:A$=""                                                    . 880
WHILE A$="":A$=INKEY$:WEND                                      . 307
RETURN                                                          . 357
PREGUNTA:                                                       . 663
INPUT "NOMBRE DEL FICHERO";A$                                   . 814
RETURN                                                          . 357
ELECCION:                                                       . 231
MENU OFF:MOUSE OFF                                              . 350
MENX=MENU(0):MENY=MENU(1)                                       . 980
ON MENX GOSUB SISTEMA,EDICION,DEFINIR                           . 347
MENU ON:MOUSE ON:RETURN                                         . 257
EDICION:                                                        . 394
ON MENY GOSUB BORRA,RESTAURA,INVERSION,INVER                    . 820
RETURN                                                          . 357
BORRA:                                                          . 516
CLS:GOSUB CUADRICULA                                            . 704
FOR I=0 TO 15:FOR IN=0 TO 15:COL(IN,I)=0:NEXT IN:NEXT I:        . 837
RETURN                                                          . 357
DEFINIR:                                                        . 985
ON MENY GOSUB PUNTERO,PUNTO                                     . 325
RETURN                                                          . 357
DESCIFRA:                                                       . 497
PO=0:FOR N=0 TO 31:GOSUB PEE3:FOR F=1 TO 8:AL=INT(N/2):         . 817
IF A1$(F)="0" AND A2$(F)="0" THEN COL(AL,((N-AL*2)*8)+F-1). 826
=0                                                              . 328
IF A1$(F)="0" AND A2$(F)="1" THEN COL(AL,((N-AL*2)*8)+F-1). 650
=2                                                              . 342
IF A1$(F)="1" AND A2$(F)="0" THEN COL(AL,((N-AL*2)*8)+F-1). 875
=1                                                              . 335
IF A1$(F)="1" AND A2$(F)="1" THEN COL(AL,((N-AL*2)*8)+F-1). 699
=3                                                              . 349
NEXT F:                                                         . 770
NEXT N                                                          . 108
RETURN                                                          . 357
PEE3:                                                           . 339
PAT=128:FOR F=1 TO 8:A1$(F)="0":A2$(F)="0"                      . 787
IF VA(AL*4+(N-AL*2))>=PAT THEN VA(AL*4+(N-AL*2))=VA(AL*4+(. 175
N-AL*2))-PAT : A1$(F)="1"                                       . 260
IF VA(2+AL*4+(N-AL*2))>=PAT THEN VA(2+AL*4+(N-AL*2))=VA(2+. 805
AL*4+(N-AL*2))-PAT:A2$(F)="1"                                   . 716
PAT=PAT/2:NEXT F:RETURN                                         . 751
INICIALIZA:                                                     . 707
WINDOW 2,"ESPERA UN MOMENTO",(30,30)-(400,60),15                . 384
FOR N=0 TO 63:VA(N)=ASC(FIL$(N+31)):NEXT N                      . 745
GOSUB DESCIFRA                                                  . 534
WINDOW CLOSE 2                                                  . 231
GOSUB LOD                                                       . 618
RETURN                                                          . 357

Numero de lineas:  269
```

```
DIM CURSOR(3,63):VALOR=1:VA=1                                   . 87
GOSUB CARGA                                                     . 994
ON MOUSE GOSUB MUEVE:MOUSE ON                                   . 349
MAIN:                                                           . 951
GOTO MAIN                                                       . 527
CARGA:                                                          . 972
FOR L=0 TO 3:NOMBRE$="CURSOR"+STR$(L+1)                         . 196
OPEN NOMBRE$ FOR INPUT AS 1                                     . 638
FOR F=0 TO 63                                                   . 777
CURSOR (L,F)=ASC(INPUT$(1,1))                                   . 984
NEXT F:CLOSE:NEXT L                                             . 582
MUEVE:                                                          . 219
MOUSE OFF                                                       . 899
VALOR=1                                                         . 402
IF MOUSE(0)>=0 THEN MUEVE                                       . 934
X=MOUSE(1):Y=MOUSE(2)                                           . 27
IF X>320 THEN VALOR=VALOR+1                                     . 225
IF Y>120 THEN VALOR=VALOR+2                                     . 192
IF VA<>VALOR THEN VA=VALOR:FOR N=0 TO 63:POKE 3204+N,           . 891
CURSOR(VA-1,N):NEXT N                                           . 939
MOUSE ON                                                        . 828
RETURN                                                          . 357

Numero de lineas:  22
```

TREPA CON

Por Santiago Estrada



Si no la conocen todavía, ya es hora de que incluyan en su vocabulario esta palabra, REXX, que seguramente procede de la latina 'REX', en castellano, REY.

NA-REXX



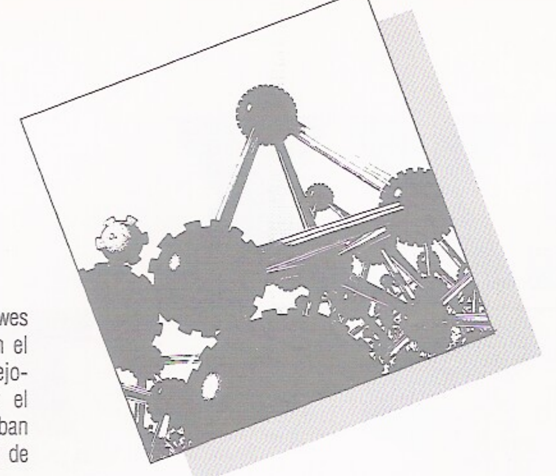
Todo el mundo en la llamada comunidad del Amiga (fuera de nuestras fronteras) habla ya del REXX. El que reciba o compre revistas extranjeras, comprobará que no se puede leer una revisión de un producto sin que se indique si soporta o no soporta REXX. Parece ya confirmado que la mismísima Commodore lo incluirá, sin cargo adicional como el AmigaBasic, en la versión 1.4 del sistema operativo.

Y es que el REXX, al igual que su casi tocayo el tiranosaurio REX del período cretácico, es el nuevo dominador de la selva multitarea del Amiga. Yo diría en broma que es el dominador común de todas las tareas, pues todas, desde el más humilde y sencillo editor de fuentes hasta el más altivo y complejo CAD/CAM 3D, se someten a los comandos de este oscuro monarca de las sombras que permanece la mayor parte del tiempo oculto en la memoria del Amiga.

El REXX es, fundamentalmente, un lenguaje de programación a mitad de camino entre el BASIC y el C, que se diseñó en principio para grandes ordenadores por Michael Cowlishaw.

Es un lenguaje especializado en el manejo de tiras de caracteres (strings) y en el reconocimiento de comandos y sus parámetros que, en definitiva, es una potente herramienta para la construcción de nuevos comandos a partir de los comandos de otras utilidades. A un comando construido a partir de otros comandos se le suele llamar 'MACRO' y a los programas AREXX se les suele llamar, por extensión, MACROS (el término 'macro' procede de los ensambladores donde una macro es una nueva instrucción definida a partir de otras instrucciones).

La implementación en el Amiga, es decir, el 'AREXX' o, para entendernos, el 'Amiga REXX', se ha llevado a cabo por William S. Hawes, muy popular por ser el creador, entre otras cosas, de ConMan y WShell. Parece, por cierto, que to-



das las cosas que hace Mr. Hawes terminan siendo un estándar en el Amiga: la mayor parte de las mejoras al 'CLI' incorporadas por el 'Shell' del 1.3, ya se encontraban en los productos antes citados de Mr. Hawes.

Muchos de ustedes tendrán ya experiencia con lenguajes de macros, ya que muchas utilidades como procesadores de texto, hojas de cálculo e incluso programas de animación, llevan incorporado un lenguaje de MACROS propio. El mismo AmigaDos contiene el comando 'EXECUTE' y sus satélites ('IF', 'LAB', 'SKIP', etc.) que sirven para ejecutar, con un solo comando, una secuencia de otros comandos (o sea una MACRO o, en términos de AmigaDos, una 'script de execute'). El disco EXTRAS del sistema operativo contiene, desde la versión 1.3, un buen 'editor de fuentes', el EMACS, que incorpora un lenguaje de MACROS muy simple pero muy útil.

Otro lenguaje de programación más, se preguntarán todos los que, como yo mismo, están ya un poco hartos del BABEL de lenguajes informáticos. Pues si, un lenguaje más, pero con una serie de características que lo hacen único y, para nosotros, los usuarios del Amiga, de incalculable valor en mi opinión.

Vamos a ver cuáles son las características que han conferido al AREXX el honor de codearse de igual a igual con el omnipresente BASIC en el futuro disco EXTRAS de la versión 1.4 (si es que su regia majestad, el AREXX, no exige un disco exclusivo para él y su amplia corte de librerías y ejemplos).

Lo que, en primer lugar, convierte al AREXX en único es que define una 'interfaz' muy sencilla (basada en el sistema de 'mensajes' y 'puertos' del EXEC del Amiga) para que todos los productos que así lo deseen comuniquen con él.

Si un programador de un producto desea que éste pueda comunicar con AREXX, solamente tiene que utilizar unas 'estructuras' y 'funciones' en la misma forma en que se usa la 'graphics.library' o la 'intuition.library'. A partir de ese momento, el producto podrá 'lanzar' o 'ejecutar' MACROS de AREXX y recibir comandos desde ellas.

Es decir, lo que convierte a AREXX en 'único' es eso precisamente, aunque resulte 'perogrullesco', que sea único. Su poder crece en la medida en que todos los productos que quieran incorporar un lenguaje de MACROS utilicen su 'interfaz' en lugar de dedicarse a construir un nuevo lenguaje.

De ello se derivan, a mi entender, las siguientes ventajas:

1. Los usuarios sólo tenemos que aprendernos un único lenguaje de MACROS en lugar de uno distinto para cada producto.

2. Los programadores se pueden concentrar en diseñar su producto y olvidarse de diseñar otro lenguaje de macros (reinventar la rueda) para que su producto sea programable y versátil. La consecuencia es que se pueden obtener productos de mayor calidad en menos tiempo y, por lo tanto, con menor coste.

3. Al reunir AREXX todos los posibles lenguajes de MACROS en uno sólo ocupa, lógicamente, menos espacio que todos ellos por separado. De esta forma se consigue que la ocupación en memoria y el tiempo de carga de los productos que usen AREXX será menor que si incorporara cada uno su propio lenguaje de MACROS (a mí me ocupa menos ya, en conjunto, que toda la parafernalia de comandos del EXECUTE, tanto en disco como en memoria).

4. Uno de los defectos de los lenguajes de macros que incorporan los productos es que suelen ser muy simples: los hay sin instrucciones de control de flujo y, si las tienen, les faltan las variables, o viceversa. El lenguaje de MACROS del AmigaDos, por ejemplo, no tiene ciclos (hay que simularlos) y, hasta la versión 1.3, no tenía variables, pero las que ahora tiene sólo se pueden usar en los comandos 'IF' y 'MORE' (con esto no pretendo decir que sea malo: algunos son peores). El AREXX, sin embargo, debido a que sus creadores se han concentrado en la tarea específica de hacer un lenguaje de macros, es tan potente que, en mi opinión, iguala a la de muchos lenguajes de programación tradicionales.

5. AREXX puede ser, para los programadores, una especie de generador de 'prototipos' y 'juegos de ensayo' que puede facilitar mucho el desarrollo y la implementación final de los productos. La consecuencia es, otra vez, el consabido 'bueno, bonito, barato'.

Supongo que a estas alturas, si he logrado su interés, querrán ver algún ejemplo. En el directorio 'EXAMPLES' del disco en el que se distribuye AREXX hay muy buenos ejemplos, pero, en este artículo de introducción, prefiero presentar un ejemplo que, no siendo demasiado complejo, permita hacerse una idea de la potencia del lenguaje.

Para ello he escogido una versión reducida de una MACRO hecha por mí mismo y que uso con bastante frecuencia (la que yo uso tiene más funciones, pero es demasiado compleja para primer ejemplo). Esta MACRO, además, envía comandos a algo que tiene todo el mundo que posee un Amiga: el AmigaDos. Para que comprendan lo que hace la MACRO primero explicaré el problema que trata de solucionar.

El AREXX es, fundamentalmente, un lenguaje de programación a mitad de camino entre el BASIC y el C.

Muchas veces necesitamos realizar un comando de AmigaDos (Copy, Delete, More, Rename, etc.) en un conjunto de directorios y ficheros. Cuando los nombres de esos ficheros tienen alguna característica común (empiezan o terminan por los mismos caracteres, etc.) podemos utilizar los 'modelos' o 'patrones' del AmigaDos que son muy potentes y que, ahora, en la versión 1.3, con los 'scripts' SPAT y DPAT del directorio 'S' del disco WorkBench podemos emplear con casi cualquier comando.

El problema empieza cuando queremos realizar un comando sobre un grupo de directorios o ficheros cuyos nombres no tienen ningún 'modelo' de caracteres en común.

Una solución consiste en utilizar el WorkBench seleccionando en los 'Drawers' los 'Proyects' y 'Tools' que deseemos pero, desgraciadamente, hasta la versión 1.4, el WorkBench sólo maneja ficheros con un '.info' asociado y, por otro lado, el WorkBench no realiza todas las funciones del AmigaDos.

Otra solución consiste en emplear una utilidad que recorra, transverse o, literalmente, 'trepe' por el árbol de directorios preguntadonos, para cada fichero, si queremos ejecutar un comando sobre él. Esto último es lo que hace, por ejemplo, el comando 'DIR' de AmigaDos con la opción I (de Interactivo), pero solamente nos permite ejecutar, en cada fichero, los comandos 'TYPE' y 'DELETE'.

Al igual que 'DIR', todas las utilidades que conozco, siempre les falta alguna que cosa que yo deseo hacer: si copian, no muestran, si muestran, no copian y, si muestran y copian, ni renombran, ni elimi-

AREXX puede ser, para los programadores, una especie de generador de 'prototipo'.

ran, pero, en caso de que hagan de todo (como Atree o Sid), apenas me caben en mis 512K.

Por lo cual, cuando compré mi versión de AREXX, me decidí a hacer una MACRO que emulara el comportamiento del 'DIR' interactivo haciendo de todo. La MACRO se llama 'TREPA.REXX' porque sirve para trepar por el árbol de directorios del AmigaDos y, para cada fichero que encuentra, nos pregunta si queremos tratarlo o no.

TREPA.REXX se usa desde el CLI o SHELL ejecutando un comando con el siguiente formato:

rx trepa <comando> <árbol> <opciones>

donde

'rx' es el lanzador, desde AmigaDos, de MACROS AREXX.

'trepa' es el nombre de la MACRO

<comando> es el comando a ejecutar (primer parámetro obligatorio)

<árbol> es el árbol (directorio o fichero) a trepar (segundo parámetro opcional, si no ponemos nada se toma el directorio actual)

<opciones> son las opciones del comando a ejecutar si las tiene (tercer parámetro opcional).

Por ejemplo, si quisiéramos recorrer el directorio del volumen introducido en 'DF0', copiando selectivamente ficheros al disco 'ram:', podríamos ejecutar el siguiente comando:

rx trepa copy df0: to ram:

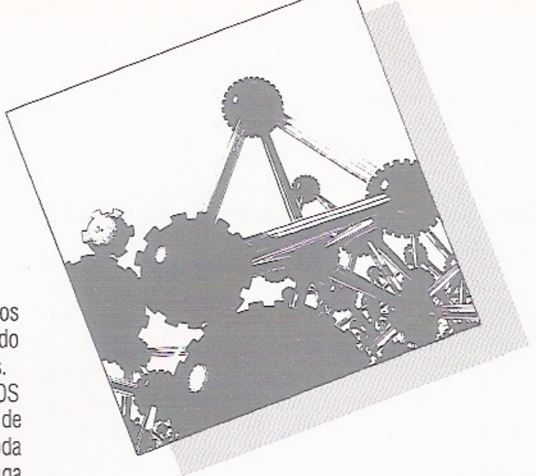
En este caso concreto la MACRO comenzará a 'trepar' por el Arbol de directorios de 'df0:' y, para cada fichero, nos preguntará si queremos copiarlo al disco 'ram:'.

Supongo que el listado de TREPA.REXX adjunto parecerá un galimatías, pero eso sucede con todos los lenguajes cuando no se conocen. Los comentarios al listado están pensados para personas que tengan alguna experiencia en programación. No recomiendo su lectura a personas que no tengan esa experiencia, ya que están pensados para comparar, en la medida de lo posible, el AREXX con el BASIC y el C y, en ningún caso, pretenden servir para enseñar AREXX.

Todo lo dicho hasta ahora no explica suficientemente porqué AREXX es una candidato para el disco de EXTRAS de la versión 1.4 del sistema operativo. Lo que, en mi opinión, hace que el AREXX sea tan útil es que puede convertirse en una 'central de comunicaciones' entre todas las tareas del AMIGA que soporten su interfaz.

De la idea de un lenguaje de MACROS independiente de cualquier producto a la idea de un lenguaje de MACROS que comunique simultáneamente con todo los productos en una máquina multitarea no hay mucho trecho que recorrer (una vez inventado parece fácil).

El AREXX, como central de comunicaciones, se encarga de recibir diligentemente los comandos,



los distribuye a sus destinatarios para que los ejecuten y, atendiendo al remite, envía las REXXpuestas.

Para comunicarse, las MACROS de AREXX utilizan el concepto de HOST. En cada momento, toda MACRO tiene una tarea del Amiga que es su HOST. A esta tarea HOST es a la que, si no se especifica otra cosa, se envían los comandos. Una MACRO puede realizar, por ejemplo, la siguiente secuencia:

1. Recibir, junto con unos parámetros, un comando del HOST actual.

2. Cambiar de HOST y, de acuerdo con el contenido de los parámetros, enviarle un comando.

3. Recibir una respuesta del nuevo HOST para, por último, volver al HOST inicial e indicarle el resultado de su comando.

Las secuencias de intercambio entre los HOST y la MACRO pueden llegar a ser todo lo complejas que se quieran.

Un HOST, para los entendidos, es cualquier 'Port' del sistema de mensajes del EXEC definido para comunicarse con AREXX. Cuando una tarea, desde un 'Port', lanza una MACRO (con un mensaje al monitor de AREXX), se convierte automáticamente en el HOST implícito o por defecto.

Si no entienden lo anterior no se preocupen, porque vamos a ver un ejemplo.

No será extraño que una vez que el AREXX esté instalado en el AMIGA el procesador de textos le pida 'favores' a la hoja de cálculo para que realice tareas que él no puede hacer.

Si estuviéramos, por ejemplo, escribiendo con un procesador de texto un informe de previsiones de gasto para el mes de febrero, necesitaremos insertar en el informe los resultados que nos da una hoja de cálculo.

Para insertar el informe de la hoja de cálculo en el texto tendríamos, como poco, que cambiar a la ventana de la hoja de cálculo, teclear los comandos para cargar la hoja de previsiones, calcularla y escribir el resultado en el disco 'RAM:' o 'RAD:', cambiar de nuevo a la ventana del procesador de texto y, por último, insertar el fichero generado con un comando u opción del procesador de texto.

Si, en cambio, tuviéramos AREXX y nuestro procesador de texto y hoja de cálculo soportaran su interfaz, lo único que tendríamos que hacer es ejecutar, desde el procesador de textos, el comando 'PREVISIONES 2' para que, en poco tiempo, después de un par de

AREXX es la respuesta del Commodore Amiga al HYPERTEXT del Apple.

gruñidos de la disquetera, apareciera la previsión de gastos del mes de febrero insertándose línea a línea en nuestro texto.

Para ver cómo se haría una MACRO que haga esto posible (con un procesador de textos y hoja de cálculo ficticios) se puede examinar el segundo listado con el fuente de la macro PREVISIONES.REXX y sus comentarios.

No sé si el anterior ejemplo da una idea exacta de la potencialidad de un mensajero programable capaz de comunicar cualquier tarea con otra en el sistema. Pero dejando volar la imaginación, las posibilidades que se abren son infinitas.

Hay quien dice, por ejemplo, que el AREXX es la respuesta del Commodore Amiga al HYPERTEXT del Apple Macintosh, calificándolo de auténtico HYPERMEDIA ya que, con AREXX, no solamente podemos referenciar texto a través de un texto, sino cualquier medio a través de otro (texto con sonidos, imagen y animación).

El AREXX es como un gurú (me refiero a un 'maestro' y no al abominable mensaje que todos padecemos) que puede sacar lo mejor que cada uno lleva dentro: el procesador de texto que procese texto, el programa de dibujo que dibuje, la hoja de cálculo que calcule o, como se suele decir, «zapatero a tus zapatos».

Creo que la capacidad multitarea del Amiga, junto con su capacidad de expansión (sobre todo en memoria) y el broche final del AREXX, precipitará el final de la era de esos 'paquetes integrados' (esos 'BRONTOSAURIOS' enormes y lentos) que hacen de todo y todo lo hacen mal: el AREXX los exterminará.

Opino también que AREXX causará la bendita desaparición de todos esos lenguajes de macros que proliferan por todas las utilidades al objeto de hacerlas programables y versátiles, pero que terminan haciéndolas innecesariamente complejas, lentas y enormes.

El mejor ejemplo que conozco de lo anterior es la utilidad de base



CLAVES A-REXX

(Para facilitar la comprensión todas las palabras clave de AREXX estan en mayúsculas pero no es necesario que sea así).

La MACRO está basada en una función 'recursiva' (que se utiliza a si misma) ya que un árbol es una estructura de datos que se define de forma recursiva: un árbol es, bien una estructura vacía, o bien un número finito de estructuras de árbol denominadas 'subárboles' del árbol. No voy a teorizar más sobre el tema, que supongo conocido, y paso a comentar las líneas del listado de TREPA.REXX.

— En AREXX todo lo que está encerrado entre '/*' y '*/', al igual que en 'C', es un comentario. Tradicionalmente las MACROS de AREXX (como los programas BASIC) comienzan por un comentario que sirve para documentar las tareas que realiza, pero en AREXX esto es algo más que una tradición: si la primera línea del fichero no es un comentario, el intérprete no lo identificará como una MACRO de AREXX.

— Con la instrucción ARG recogemos, con facilidad pasmosa, los parámetros que hemos pasado en la llamada a la MACRO separados por espacios (esto se explica más adelante a propósito de la instrucción PARSE).

Notar que en AREXX, al igual que en AmigaBasic y al contrario que en C, no se declaran previamente las variables. Eso sí, hay dos diferencias con el BASIC: primero, si una variable no está inicializada su valor no es nulo como en BASIC, sino que toma el propio nombre de la variable y, segundo, las variables no tienen 'tipo', el AREXX no distingue entre 'enteros','reales','tiras', etc., para el AREXX, en realidad, todos son tiras, y, si una tira contiene un número correcto, entonces puede tratarse con los operadores matemáticos.

— Llamamos a una función, 'STATEF', de la 'rexxsupport.library' que nos devuelve una tira con las características del árbol (si es directorio o fichero, octetos y/o bloques que ocupa, el comentario de 'FILENOTE', etc.) y extraemos de esa tira la primera palabra con la función WORD para almacenarla en la variable TIPO (la primera palabra es 'DIR' si el árbol es un directorio, 'FILE' si es un fichero y nada si no existe).

— La instrucción SELECT es igual a la SWITCH de 'C' y su equivalente en AmigaBasic es un IF con ELSEIF's. Con la SELECT, de acuerdo con el valor de Tipo, nos dirigimos a trepar por un directorio llamando a la función recursiva Trepar, a tratar un fichero con la función Tratar o, si no existe el árbol pedido en la llamada, a enviar un mensaje de error al usuario con la instrucción SAY que cumple las mismas funciones que el PRINT de BASIC o el PRINTF de C.

A la vuelta de las funciones llamadas, el árbol pedido se ha tansversado o 'trepado' entero y no queda nada más por hacer: con la instrucción EXIT terminamos la MACRO.

Incidentalmente, como curiosidad, notar que hay que tratar de forma diferente el primer nodo del árbol de directorios debido a que puede ser una especificación de volumen (nombre seguido de ':'). A reconocer y tratar esta eventualidad están dedicadas las líneas que utilizan la función RIGHT muy parecida al RIGHT* de basic.

Notar también la cómoda forma en que trata el AREXX la concatenación de 'strings': si dos tiras están separadas por un blanco AREXX las concatena con un espacio intermedio; si están concatenadas con el operador '||' las concatena sin espacio intermedio (igual que el operador '+' de AmigaBasic). Es decir:

SAY 'separado por' 'un espacio'
visualizaría la siguiente línea en la consola:
separado por un espacio
En cambio la siguiente instrucción
SAY 'todo' || 'junto'
visualizaría la siguiente línea:
todo junto

— La función trepar comienza en la línea con la etiqueta (igual a las de AmigaBasic) 'Trepar:'.

La instrucción 'PROCEDURE' indica que la función es un como un 'SUBPROGRAM' de AmigaBasic o una 'FUNCTION' de C, es decir, las variables que nombremos dentro de la función no 'colisionan' con las ya nombradas en otras partes de la MACRO o, para entendernos, las variables son locales a la función: si utilizamos el mismo nombre de variable en el programa principal y en la función, el AREXX no los considerará la misma variable, a menos que la mencionemos en la lista que sigue al parámetro EXPOSE (que cumple igual función que el STATIC de AmigaBasic o el EXTERN de C).

Sin la instrucción PROCEDURE una función de AREXX es igual que el objeto de un GOSUB de BASIC (sin equivalente en C).

La instrucción ARG hace lo mismo en este caso que en el anterior: recoge los parámetros de la llamada a la función que, ahora, solamente es el 'subárbol' al que hay que 'trepar'.

— Se llama a la función Pedir (que comento en el punto 10) para saber si el usuario desea entrar o 'subir' al 'subarbol' presente; la respuesta del usuario se almacena en la variable Respuesta. Si la respuesta es distinta de 'S' Trepar retorna a la función que la llamó (ella misma o la MACRO principal), devolviendo el valor de Respuesta. Si la respuesta es 'S' entonces utilizamos la función de SHOWDIR de la librería de soporte que nos devuelve una lista de los ficheros y subdirectorios de un directorio separados, en este caso, por el carácter '/'.

— La instrucción DO FOREVER itera sin fin, ejecutando las instrucciones comprendidas entre ella y su correspondiente END. Es equivalente a una sentencia, por ejemplo, FOR (;;) en C (de la que hay que salir con un BREAK, RETURN o GOTO) o a una WHILE -1 en AmigaBasic (de la que podemos también salir con GOTO o RETURN).

— Dentro del cuerpo de la instrucción DO encontramos, en primer lugar, la misteriosa instrucción PARSE. Esta no tiene equivalentes en AmigaBasic o C y es una de los potentes herramientas de AREXX para manejar comandos y sus parámetros. ▶

El MicroFiche Filer Plus pone el acento sobre la visualización de los datos y no sobre su almacenamiento.

```
/* Trepa a un Arbol AmigaDos ejecutando un comando selectivamente
*/ (1)
ARG Comando Arbol Opciones  (2)
Tipo=WORD(STATEF(Arbol),1)  (3)
SELECT                      (4)
WHEN Tipo=='DIR' THEN
  IF RIGHT(Arbol,1)==':' | RIGHT(Arbol,1)=='/' THEN
     CALL Trepar Arbol
     ELSE CALL Trepar Arbol || '/'
WHEN Tipo=='FILE' THEN CALL Tratar Arbol
OTHERWISE SAY 'No encuentro el arbol' Arbol 'pedido'
END
EXIT

Trepar: /* trepar al Arbol */ (5)
PROCEDURE EXPOSE Comando Opciones
ARG Arbol
Respuesta=Pedir('Entrar en directorio:' Arbol) (6)
IF Respuesta ~=='S' THEN RETURN Respuesta
Lista=SHOWDIR(Arbol,'A','/')
DO FOREVER                                     (7)
   PARSE VAR Lista Nombre '/' Lista            (8)
   IF Nombre=='' THEN
      DO
        SAY 'Terminado el directorio' Arbol 'Retrocediendo ...'
        RETURN 'N'
      END
   ELSE
      DO
        Tipo=WORD(STATEF(Arbol || Nombre),1)
        SELECT
        WHEN Tipo=='DIR' THEN
           IF Trepar(Arbol || Nombre || '/')=='R' THEN RETURN 'N'
        WHEN Tipo=='FILE' THEN
           IF Tratar(Arbol || Nombre )=='R' THEN RETURN 'N'
        OTHERWISE SAY 'No encuentro el Arbol ' Arbol || Nombre
        END
      END
END
/* Nunca devuelve control por aqui */

Tratar:                          (9)
PROCEDURE EXPOSE Comando Opciones
ARG Arbol
Respuesta=Pedir('Desea ejecutar' Comando Arbol Opciones)
IF Respuesta=='S' THEN ADDRESS COMMAND Comando Arbol Opciones
RETURN Respuesta

Pedir:                           (10)
ARG Mensaje
DO FOREVER
   SAY Mensaje
   SAY 'Conteste con una abreviatura de
Terminar/Retroceder/Si/No'
   SAY 'Si pulsa retorno se tomar "NO" por defecto'
   PULL Respuesta .
   SELECT
   WHEN ABBREV('TERMINAR' ,Respuesta,1) THEN
        DO
          SAY 'Terminado el recorrido '
          EXIT
        END
   WHEN ABBREV('RETROCEDER',Respuesta,1) THEN RETURN 'R'
   WHEN ABBREV('SI' ,Respuesta,1) THEN RETURN 'S'
   WHEN ABBREV('NO' ,Respuesta,0) THEN RETURN 'N'
   OTHERWISE SAY 'Entre una respuesta de las indicadas, gracias'
   END
END
```

La instrucción PARSE provee un mecanismo para extraer una o más subtiras de una tira y asignarlas a variables. La tira de entrada puede venir de varias fuentes: una expresión, la llamada a una MACRO o función (la instrucción ARG es en realidad una PARSE ARG) o cualquier expresión o variable (como en el caso presente). La sintaxis de PARSE es la siguiente:

PARSE [UPPER] TiraFuente [Patrón] [,Patrón] ...

El parámetro UPPER, si se incluye, convierte a mayúsculas la subtiras extraídas de TiraFuente antes de trasladarlas a las variables indicadas por los patrones.

de datos MicroFiche Filer Plus. Esta utilidad es una base de datos hecha a la medida del Amiga y completamente distinta de todas las utilidades de base de datos en boga.

El MicroFiche Filer Plus pone el acento sobre la visualización de los datos y no sobre su almacenamiento y procesamiento: está hecha para ver los datos. No es el momento de hacer una revisión de MicroFiche Filer Plus, pero, en mi opinión, puede llevarse el 'Oscar' a 'la utilidad más original concebida de una sola vez' desde el advenimiento, hace ya tiempo, del Visicalc (la Eva de todas las hojas de cálculo).

La primera versión de MicroFiche Filer (sin Plus) no era programable. La versión PLUS soporta AREXX y, por lo tanto, es totalmente programable. Es decir, con una mínima ocupación en disco y memoria tenemos una utilidad, única en su género, mucho más potente que otros pesos pesados del mismo ramo que, además, es mucho más barata.

Hay un par de cuestiones que supongo de interés y que creo que no quedan suficientemente aclaradas. La primera es qué utilidad tiene el AREXX para usuarios no programadores. Para los no programadores el AREXX puede dar sus servicios de la misma forma que el lenguaje C, es decir, mediante macros hechas por otros. La diferencia con el C es que, al ser AREXX un intérprete, se necesita poseer AREXX y ejecutar su MASTER para poder utilizarlo.

La segunda cuestión es para programadores: dominando AREXX podemos olvidarnos del BASIC, podemos olvidarnos del C o de un lenguaje parecido como el Modula o el Pascal. La respuesta depende de cuáles sean nuestro objetivos de programación.

El AREXX sirve de apoyo a la programación de productos pero solamente desvirtuándolo podríamos utilizarlo para desarrollar un producto completamente con él (que no sea, claro está, un conjun-

to de MACROS de AREXX). Para los que desarrollan productos de utilidad, digamos, pública, el AREXX es solamente un complemento ideal para el producto a la hora de ejecutarlo y una ayuda a la hora de desarrollarlo.

Para aquellos usuarios que sólo programan de forma casual para hacer pequeñas utilidades de valor particular creo que AREXX es su herramienta ideal. Yo antes usaba el BASIC para esas tareas y el C para las cosas serias. Ahora uso AREXX para las cosas sencillas y, gracias a su capacidad, el ensamblador para las cosas serias. El conjunto ASSEMBLER/AREXX de ahora me resulta mucho más ágil y, a la vez, más potente e integrado que el conjunto C/AmigaBasic.

Si lo que se necesita es una 'Primera Lengua' para aprender a programar el AREXX está especialmente recomendado para ello y, además, los programadores de AREXX están bastante solicitados para trabajar en entornos de grandes ordenadores compatibles. ■



El patrón es, a la vez, una manera de indicar las variables que van a recibir las subtiras y de indicar cómo se forman estas últimas. Hay varias formas de indicar cómo se forman las subtiras, pero la principal es que una variable se llena con todos los carácteres de la tira fuente hasta que se encuentra un caracter específico. En la instrucción PARSE del ejemplo se emplea una '/' como separador para la subtira que irá a la variable Nombre, el resto de la tira fuente se lleva sobre lista.

El efecto final de la instrucción PARSE del ejemplo es que obtenemos en la variable 'Nombre' el nombre del primer fichero o subdirectorio de la lista, al mismo tiempo que quitamos de Lista este nombre dejando el resto.

Notar que la instrucción ARG empleada antes es en realidad una PARSE ARG y lleva sobre las variables que se indican las subtiras separadas por espacios de la llamada a la MACRO o la función interna.

Cuando en 'Nombre' obtengamos una tira nula después del PARSE es que se ha acabado la lista de ficheros y directorios del nodo presente y retornamos al anterior.

Si 'Nombre' contiene algo examinamos, igual que en la parte principal de la MACRO, si es fichero o directorio y procedemos, mediante una SELECT, a llamar a Trepar() otra vez si es un directorio, a Tratar() si es un fichero o a enviar un mensaje de error si el contenido de 'Nombre' no existe. Si Trepar() o Tratar() devuelven una 'R' es que el usuario desea retroceder al nodo padre del presente.

Parecerá raro que incluya una salida en la SELECT para enviar un mensaje de error cuando no exista un nombre de la lista que acabamos de obtener: en un sistema multitarea como el del Amiga uno nunca puede estar seguro de que en el intervalo, desde que se obtiene la lista hasta que se comprueba el nombre, alguna otra tarea no haya eliminado el nombre.

— La función Tratar es la que se encarga de ejecutar el comando pedido sobre un fichero. Primero pregunta al usario si desea ejecutar el comando para ese fichero y, en caso afirmativo, lo intenta ejecutar. La instrucción ADDRESS COMMAND envía la expresión formada detrás de ella al AmigaDos para que la ejecute.

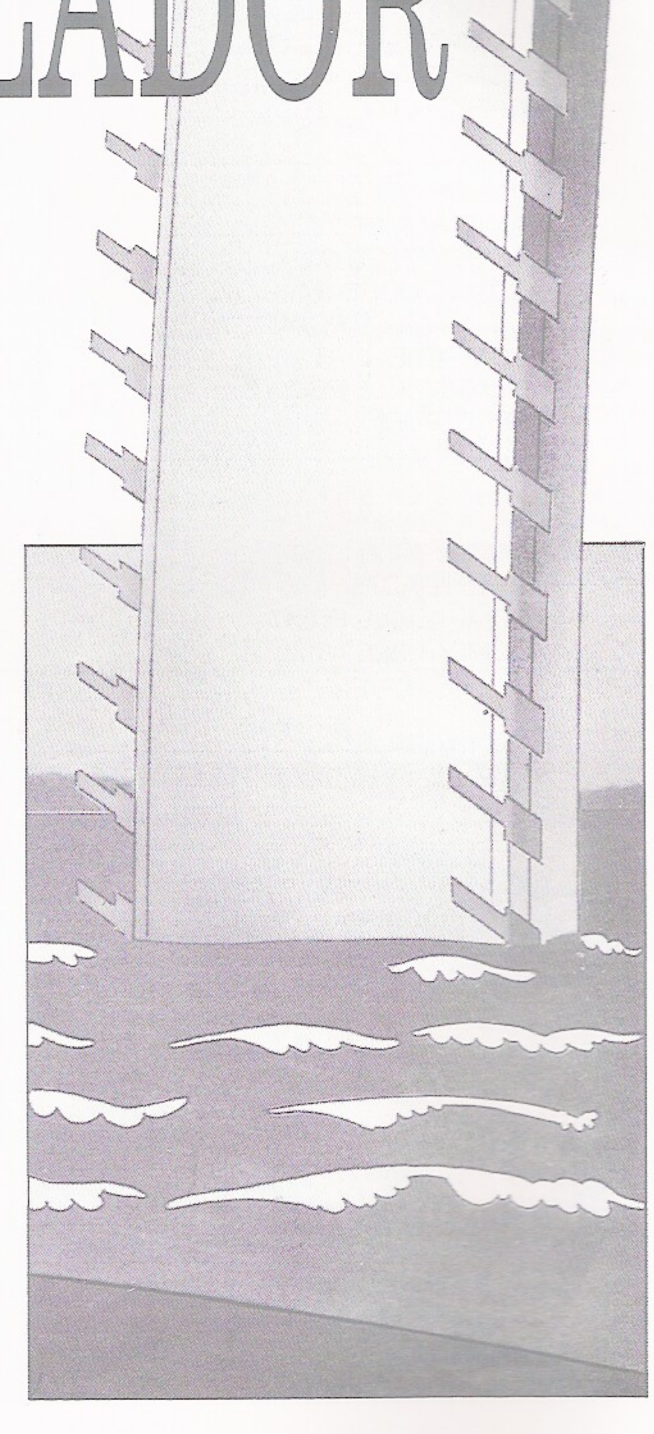
— La función Pedir no tienen mucho que comentar. Visualiza un mensaje de petición con instrucciones para el usuario que puede responder con Si, No, Terminar -que se comentan ellas solas- y Retroceder, que significa que queremos terminar con el recorrido del árbol presente y retornar al anterior.

Hay que destacar la función ABBREV que devuelve un valor Booleano cierto si la variable Respuesta contiene una abreviatura válida de cada una de las posibles contestaciones hasta el número de caracteres indicado (uno en todas, excepto en 'No', que admite la tira nula como abreviatura). ■

INICIACION AL LENGUAJE ENSAMBLADOR

SISTEMA DE FICHEROS 9.ª Parte

Por Fernando G. Terradillos



En este capítulo trataremos el sistema de distribución de información dentro de un disco. Un elemento bastante desconocido al usuario, pero que trataremos en profundidad.

Para ello veremos la organización jerárquica de bloques para su posterior lectura o grabación con cómodas rutinas. Para completar el capítulo veremos cómo funciona el bootblock, es decir, los bloques 0 y 1 de un disco ejecutable, por ejemplo, su workbench.

Comencemos por el fondo del asunto, es decir, cómo almacena el Amiga toda su información en el disco, siendo esto posible gracias a su compleja estructura, que soporta en un mismo disco un enorme número de ficheros y directorios.

Para continuar debe saber que un disco se compone de 80 cilindros (del 0 a la 79), cada uno de ellos con dos cabezas de lectura (superior e inferior) y en cada una de ellas 11 sectores (del 0 al 10, 22 en total por cada cilindro). Esto hace en total 1760 bloques (del 0 al 1.759). Cada bloque en cuestión contiene 512 bytes, siendo la totalidad exacta de un disco en bytes 901.120. Pero si sólo disponemos de 856 k. cuando formateamos uno de ellos, dónde se encuentra el resto. Pues exactamente está ocupando bloques tan importantantes como el bootblock (2 bloques = 1 k.), el directorio principal (1 bloque), el utilizado como mapa de bloques libres en disco (1 bloque), y si tenemos en cuenta que por cada bloque arranca 24 bytes al disco (un total de 42144 bytes) esto hace una suma suficiente para que nos demos cuenta que el sistema operativo del disco usa parte del mismo para utilización propia.

Resumen de organización del disco:

80 cilindros = 160 pistas.
2 cabezas = 80 pistas por cabeza.
11 sectores por pista = 1.760 bloques.
1 bloque = 512 bytes.
1.760 bloques = 901.120 bytes.

Como he dicho antes, el sistema para organizar todos estos bloques es jerárquico, como el de un árbol, teniendo como base el directorio principal, que es el que conseguimos al hacer un simple DIR desde el CLI a cualquier disco. Este directorio principal ocupa simplemente un bloque de disco, y para mayor facilidad de búsqueda se ha situado en el centro exacto del disco (bloque 880, sector 0, cabeza 0, cilindro 40). Esto es totalmente lógico, pues sería de tontos meter el directorio princial en el comienzo del disco para que el cabezal del DRIVE se volviera loco dándose unos paseos por toda su superficie.

A continuación voy a presentar los 512 bytes del bloque 880 de mi Workbench, para que a continuación explique cuál es el contenido del mismo. Antes de hacerlo quiero decir que hay un libro magnífico que recomendé en el capítulo 1 y voy a repetir su título: «The AmigaDOS Manual», de BANTAM AMIGA LIBRARY.

También recomiendo utilizar algún editor de discos (DISK ED, por ejemplo) para hacer el seguimiento poco a poco.

La información de la derecha pertenece al offset o posición dentro del bloque en hexadecimal. Los números del centro pertenecen a la notación hexadecimal de la información del bloque en sí. La parte de la derecha pertenece a la información en ASCII correspondiente a la misma línea.

Como podrá comprobar de un vistazo ya estamos viendo el nombre del disco en la parte inferior del bloque. También hemos de decir que la información de estos tipos de bloques se agrupan de cuatro en cuatro bytes (dobles palabras), excepto para la información de nombres de ficheros, directorios, etc.

El resto de la información vamos a darla como offsets en hexadecimal, es decir, por ejemplo el offset $0138 contiene $FFFFFFFF y explicaremos el contenido de la misma posición (cuadro 1).

Ahora vamos a ver cómo enlaza el directorio principal con otro directorio; en este caso pondremos el directorio 'devs' que contiene la información acerca de los periféricos del Amiga. Para encontrarlo hay que mirar las posiciones dentro de la tabla de bloques entre los offsets $0018-$0137, y es exactamente el $0070 el que enlaza al directorio. El valor contenido en este offset es el de $0000037D que en decimal se convierte a 893, siendo éste el bloque exacto que contiene la información necesaria para enlazar el directorio 'devs' con otros ficheros u otros directorios. Quiero hacer ver que todavía no hemos llegado a la información propia que contiene un fichero, sino que estamos viendo qué es lo que hace el Amigados hasta llegar a un fichero que le hemos pedido. Y a continuación veamos el bloque referente al directorio 'devs' en mi Workbench.

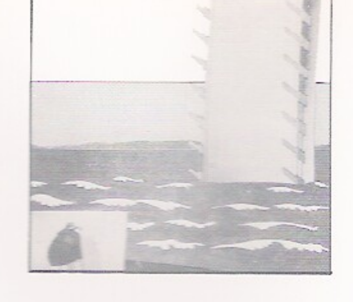
También podrá ver a primera vista el nombre del directorio, siendo, asi mismo, la estructura parecida a la del directorio principal. Veamos cuáles son las diferencias (cuadro 2).

Como podrá observar, la semejanza al directorio principal por parte de un directorio de usuario es bastante, ya que los dos son directorios. A continuación veamos el bloque referente a un fichero, en este caso al 'printer.device', el driver de la impresora.

El bloque en cuestión es el que se encuentra el el offset $0098 con el valor $402 (en decimal 1026). El bloque de cabecera de fichero contiene información parecida a los dos tipos de bloques anteriores. Veámoslo.

Podrá observar una serie más o menos consecutiva de números, éstos pertenecen a la lista de bloques en el que se encuentran los datos fichero en cuestión, asimismo contiene información ya vista anteriormente. Veamos la configuracion de este tipo de bloques (cuadro 3).

Cuando un fichero contiene más bloques de los que caben en el rango de los offsets dedicados a ellos se crea un bloque de exten-



sión que contentra la continuación de serie de bloques que le faltaban al fichero. En este caso no tiene bloque de extensión, pero veremos otro ejemplo en el siguiente tipo de bloque:

—$01F4 $0000037D, bloque apuntando al directorio anterior, exactamente a 'devs', ya que este bloque pertenece a la estructura del directorio. Observa que el número $037D es el mismo que contenia el anterior tipo.

—$01FC $FFFFFFFD, segundo número indicando el tipo de bloque. Si lo cambiamos como negativo nos encontramos con $-3, el valor exacto perteneciente al tipo de bloque.

Cuando el Amiga tiene que encontrar el nombre del fichero en cuestión para su posterior lectura, busca entre todos los bloques del directorio que tiene presente que le indica los punteros de los offsets $0018 a $0137. Cuando lo haya encontrado pasa a almacenar la cabecera del bloque que le dará toda la información vista anteriormente.

Ahora pasará a la lectura del fichero y para ello chequea el número de bloques que contiene el fichero (offset $0008), lmacena la lista de bloques consecutiva contenida en los offsets $0018 a $0137 (en sentido inverso), y si el fichero es lo suficientemente largo para que haya un bloque de expansión, también pasa a su lectura.

En el caso anterior, de que necesite un bloque de expansión para almacenar esa lista de punteros, se creará un nuevo bloque llamado simplemente de expansión de fichero. Veamos el ejemplo de mi Workbench con el fichero Preferences situado en el directorio principal.

La estructura para este tipo de bloques es el cuadro 4.

El último tipo de bloque que queda es el de bloque de datos, es decir, el que realmente contiene los datos del fichero. Como dije anteriormente, 24 bytes de cada uno de estos bloques contienen información para el sistema operativo, y que en realidad se pierden, pues el nuevo (bueno, ya un poco

CUADRO 1

OFFSET	VALOR	CONTENIDO
-$0000	$00000002	tipo de bloque, en este caso 2 = primario
-$0004	$00000000	número cabecera, siempre 0
-$0008	$00000000	número de secuencia mayor, siempre 0
-$000C	$00000048	tamaño tabla en dobles palabras (=tamano bloque-56)
-$0014	$AE6FF84C	CHECKSUM o suma de chequeo, contiene el valor $FFFFFFFF menos la suma total del resto de dobles palabras.
-$0018-$0137	---------	este rango contiene los punteros a otros bloques.

Por cada número obtenemos el enlace a un fichero o directorio.Más tarde veremos un ejemplo de cualquiera de ellos.

OFFSET	VALOR	CONTENIDO
-$0138	$FFFFFFFF	este valor es el que se consigue si el mapa de bloques libres es el correcto. Se pondrá a 0 cuando haya un error en la validación del disco.
-$013C-$01A3	---------	este rango pertenece a los bloques que contienen el bitmap (no confundir con bitmap de un gráfico) o mapa de bloques libres del disco.

Cada vez que se incluye un fichero en el disco, el AmigaDOS actualiza estos bloques para saber cuales le quedan libres para un futuro uso.

OFFSET	VALOR	CONTENIDO
-$01A4-$1AB	---------	fecha en dias y minutos en el que se vario el directorio o fichero
-$01B0-$1CF	---------	nombre del directorio principal, cuyo primer byte, en este caso $10 es el número de caracteres del nombre, con un máximo de 30.
-$01E4-$1EB	---------	fecha de creación directorio principal
-$01F0	$00000000	siguiente entrada en tabla de bloques.

Cuando el directorio principal contiene más ficheros de los que puede admitir se crea un bloque de expansión que contiene más punteros. Cuando esta a 0 es que no hay expansión, cualquier otro número indicaria que si tiene.

OFFSET	VALOR	CONTENIDO
-$01F4	$00000000	directorio anterior, que en el caso de ser directorio principal no lo posee, pues es el mayor rango dentro del disco.
-$01F8	$00000000	extensión, siempre a 0
-$01FC	$00000001	segundo número indicando el tipo de bloque

CUADRO 2

OFFSET	VALOR	CONTENIDO
-$0000	$00000002	tipo de bloque, 2 = primario
-$0004	$0000037D	cabecera, apuntándose a el mismo
-$0014	$889B5C91	CHECKSUM
-$0018-$0137	---------	tabla de bloques conteniendo punteros a los directorios y ficheros que contienen el directorio. El siguiente ejemplo lo veremos con un fichero.
-$0140	$00000005	bits de protección del directorio, correspondientes a RWED (lectura,escritura,editar,borrar), en este caso 5 en binario es 0101 y por consiguiente estan activados los bits de protección contra la escritura y el borrado.
-$148-$01A3	---------	comentario del directorio, en este caso a 0
$01A4-$1AB	---------	fecha en dias y minutos en que se creo el directorio
-$01B0-$1CF	---------	nombre del directorio, con la misma estructura de longitud que el directorio principal.
-$01F0	$00000000	siguiente entrada en tabla de bloques
-$01F4	$00000370	directorio anterior, que en este caso es el directorio principal con el valor en hexadecimal $0370 = 880. Este valor lo utiliza para una mayor agilidad de ir retrasando directorios.
-$01F8	$00000000	extensión, siempre a 0
-$01FC	$00000002	segundo número indicando el tipo de bloque, en este caso el valor 2 representa a un directorio de usuario.

CUADRO 3

OFFSET	VALOR	CONTENIDO
-$0000	$00000002	tipo de bloque. 2 = primario
-$0004	$00000402	cabecera, apuntandose a el mismo
-$0008	$00000037	número de bloques del fichero
-$0010	$00000087	primer bloque del fichero
-$0014	$EB52C434	CHECKSUM
-$0018-$0137	---------	tabla de bloques conteniendo los bloques pertenecientes al fichero, comenzando de abajo a arriba.

Si el offset anterior que indica el primer bloque del fichero es $87 la continuación de bloques seria esta: $89,$8A,$8B,$8C,$8D,$8E,$79,$7A,$7B,$7C,$7D,$7E,$7F,$80,$81,$82 y así hasta completar el número de bloques que nos indicaba el otro offset.

OFFSET	VALOR	CONTENIDO
-$0140	$00000000	bits de protección del fichero, en este caso sin protección alguna.
-$148-$01A3	---------	comentario del fichero
-$01A4-$1AB	---------	fecha en dias y minutos de la creación del fichero
-$01B0-$1CF	---------	nombre del fichero
-$01F0	$00000000	siguiente entrada en tabla de bloques.

antiguo) Fast File System lo resuelve. Para ello carga sólo en memoria toda la lista de bloques correspondiente al fichero y luego los lee todos. Esto en disco duro es enormemente favorable, consiguiendo aceleración en lectura cuatro veces superior al sistema antiguo. La estrucura para cualquier bloque de datos es el cuadro 5.

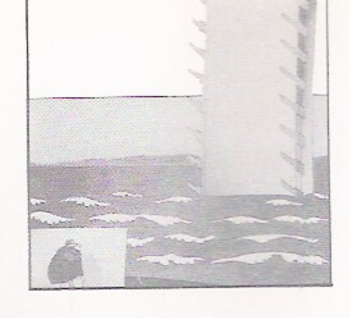
Voy a hacer un resumen viendo cómo el sistema operativo del disco lee el fichero printer.device del directorio 'devs' del Workbench. Lo primero que realiza el ordenador cuando insertamos un disco dentro del Workbench o del CLI es leer el bloque 880 para saber su nombre y que está insertado. En estos momentos el bloque es la única información que tiene el Amiga acerca del disco.

CUADRO 4

OFFSET	VALOR	CONTENIDO
-$0000	$00000010	tipo de bloque. 2 = bloque lista
-$0004	$00000453	cabecera, apuntando a el mismo
-$0008	$0000002B	número de bloques en la expansion
-$0014	$FFFF7866	CHECKSUM
-$0018-$0137	---------	expansión de lista de bloques. Si aún el fichero tiene más bloques comprobar posteriormente el siguiente offset de expansión.
-$01F4	$00000385	puntero al bloque anterior en la expansión de la lista, en este caso la cabecera del fichero.
-$01F8	$00000000	siguiente bloque con la expansion de la lista de bloques, si el fichero es aún más largo de los bloques que puede almacenar el rango.
-$01FC	$FFFFFFFD	al igual que en el bloque de cabecera.

CUADRO 5

OFFSET	VALOR	CONTENIDO
-$0000	$00000008	tipo de bloque. 8 = bloque de datos
-$0004	$00000402	cabecera, apuntándose a el mismo
-$0008	$00000016	número de secuencia, o número de posición en la lista de bloques
-$0010	$000001E8	tamano del bloque = 488 bytes. Este es un número constante en todos los bloques de datos.
-$0014	$00000072	puntero al siguiente bloque en la lectura del fichero
-$0018	$99C0C4FD	CHECKSUM
-$001C-$200	---------	datos del fichero.



LISTADO 1

```
          BSR.L    OPENL        ;ABRE LIBRERIA DOS
          BSR.L    WRITE        ;INICIALIZA ESTRUCTURAS
          BSR.L      PROG       ;PROGRAMA PRINCIPAL
          RTS             ;RETORNA AL SISTEMA

OPENL           MOVEA.L 4,A6          ;EXEC
          LEA      DOSLIB,A1    ;NOMBRE LIBRERIA
          JSR      -$198(A6)    ;ABRE LIBRERIA
          MOVE.L   D0.DOSBASE   ;GUARDA BASE LIBRERIA
          RTS



********* RUTINA PRINCIPAL: INSERTA E A1 LA ESTRUCTURA IORequest


WRITE           MOVEA.L 4,A6          ;EXEC
          SUBA.L   A1,A1        ;0 EN A1
          JSR      -$126(A6)    ;BUSCA TASK O TAREA PRESENTE
          MOVE.L   D0.TASK            ;GUARDAR PUNTERO
          LEA      DISKPORT,A1 ;APUNTAR ESTRUCTURA PORT
          MOVE.L     D0,$10(A1)       ;INSERTA TASK EN DISKPORT
          JSR      -$162(A6)    ;PONER PUERTO
          LEA      IOREQ,A1     ;ESTRUCTURA IO REQUESTER
          MOVEQ    #0,D0        ;UNIDAD
          MOVEQ    #0,D1        ;BANDERAS
          LEA      DEV,A0       ;NOMBRE PERIFERICO
          LEA  DISKPORT,A2      ;INSERTA DISKPORT EN IOREQ
          MOVE.L     A2,D2            ;
          MOVE.L     D2,$0E(A1)       ;
          JSR      -$1BC(A6)    ;ABRIR PERIFERICO
          RTS



******************** PROGRAMA: LECTURA BLOQUE 880


PROG      MOVEA.L 4,A6          ;RUTINA DE LECTURA DE BLOQUES
          MOVE.W  #2,$1C(A1)    ;2=READ
          MOVE.L  #$6E000,$2C(A1)   ;OFFSET
          MOVE.L  #$200,$24(A1)     ;LONGITUD
          MOVE.L  #$60000,$28(A1)   ;POSICION DE LECTURA
          JSR     -$1C8(A6)     ;DoIO DE LECTURA
          MOVE.W  #4,$1C(A1)    ;4=UPDATE=ESCRIBIR TODO BUFFER
          JSR     -$1C8(A6)     ;DoIO
          MOVE.W  #9,$1C(A1)    ;9=NONSTD=ERROR=PARA MOTOR
          MOVE.L  #0,$24(A1)    ;LONGITUD
          JSR     -$1C8(A6)     ;DoIO
          RTS



******************** FIN PROGRMA, COMIENZO PUNTEROS


          CNOP 0,2
DOSBASE         DC.L 0
IOREQ           DS.B $30;DISKPORT   DS.B 22
TASK      DC.L 4


          CNOP 0,2
DOSLIB          DC.B 'dos.library',0
          CNOP 0,2
DEV       DC.B 'trackdisk.device',0

          END
```

Ahora para pasar al directorio 'devs' dentro del CLI entraremos el comando ya conocido de CD seguido del nombre. El amiga busca entre la tabla de bloques conteniendo punteros hacia todos los directorios y ficheros que se encuentran en el bloque 880 o directorio principal.

Una vez que ya lo ha encontrado pasa a leer en memoria la cabecera del directorio encontrado. Dentro de éste hay otra lista de bloques apuntando a otros ficheros y directorios, siendo en este caso el fichero printer.device. Al igual que en el directorio principal, el Amiga busca de la misma manera y una vez que la encuentra lee el bloque en memoria.

Este bloque se trata de la cabecera del fichero, conteniendo otra lista, que le sirve para leer todos los bloques pertenecientes a los datos reales del fichero. En muchos casos, ocupan más número bloques del los que se puede almacenar en la cabecera del fichero, creándose bloques de extensión.

Para leer el fichero, el Amiga carga en memoria todos los punteros de los bloques de datos, y a continuación los carga uno a uno, es decir, comprobando que esos punteros corresponden con los que se encuentran dentro de cada bloque de datos. En Fast File System esto no lo comprueba, siendo su acceso a los datos muy rápida.

A la hora de grabar un fichero el sistema que utiliza es sencillo. Cada disco tiene un mapa de bloques libres. Primero graba el puntero del fichero dentro de la lista del directorio en el que se encuentra en el momento. Actualiza el mapa de bloques y por último graba todos los bloques de datos.

Esto último tiene un enorme inconveniente, pues sería lo más lógico actualizar el mapa de bloques libres en último lugar. Si ocurre algún gurú miestras estamos grabando un fichero aparte de que nos hemos quedado sin fichero, el sistema de validación (reactualizar el mapa de bloques libres) falla con gran asiduidad, teniendo que copiar todos los ficheros de nuevo a otro disco. Mala suerte para los que posean un disco duro original de Commodore (no he podido probar otros sistemas), pues formateándolo en FFS, si grabando un fichero se queda colgado el Amiga, va a tener grandes problemas si al arrancar de nuevo se encuentra que tiene todos los bloques disponibles, aunque estén presentes todos los ficheros. Esto es debido al incorrecto sistema de validación, el cual se podría resolver si a la hora de validar el disco, por cualquier error, verificara la presencia de cada fichero reactualizando por cada uno el mapa de bloques libres. Ahora pasemos a otro tema, que es el misterioso y desconocido bootblock, que tantos quebraderos de cabeza ha traído, pues son los portadores de la mayoría de los virus. El bootblock en el Workbench tiene la misión de configurar todo el sistema (memoria, librerías, periféricos, etc.) para luego activar un CLI, cargar las preferencias con el fichero system-configuration del directorio 'devs': y ejecutar el fichero startup-sequence del directorio.

Técnicamente el bootblock son los bloques 0 y 1 de cada disco. Cuando insertamos el disco en el ordenador al empezar la sesión el Amiga carga estos dos bloques (1.024 bytes) en memoria de zona alta en Chip Memory. Las tres primeras largas palabras (12 bytes) son fundamentales para que el ordenador se dé cuenta de que es un disco ejecutable. La primera de ellas contiene el texto DOS seguido del byte 0, que indica el tipo de formato que lleva el disco. Los segundos cuatro bytes son el CHECKSUM o suma de chequeo, en el cual resta el valor hexadecimal $FFFFFFFF a la suma de todas las restantes largas palabras, para luego compararlos. Si no corresponden con el valor no continuará el proceso. La tercera palabra larga contiene el bloque donde se encuentra el directorio principal.

En el momento de que el sistema cargue estos dos bloques insertará dos valores en dos registros. En A0 pondrá la dirección de ejecución (posición $C = 12 a partir del bootblock) y en D0 pondrá un valor 0 si el proceso anterior del chequeo de suma ha salido correcto. Una vez verificado todo esto el resto de 1.012 bytes contiene el programa que el ordenador ejecuta. En el momento de hacerlo el sistema tiene un registro ajustado para que lo utilice el usuario. Este es el A1, en el que apunta a una estructura IORequest perteneciente al disco, es decir la estructura encargada de realizar todos los procesos que lleva un drive (lectura, grabación, etc., de bloques). Resumiendo, el ordenador se enciende, insertamos nuestro Workbench o juego preferido, el sistema carga en una posición de zona alta de Chip Memory los bloques 0 y 1 (bootblock). Comprueba que los primeros cuatro bytes llevan el texto DOS, los segundos cuatro bytes se ajustan a la suma de chequeo y que la tercera palabra larga contiene el valor $370 correspondiente al bloque 880 o que

es lo mismo, directorio principal. Una vez comprobado inserta en el registro A1 un puntero a la estructura IORequest del 'drive' y ejecuta el código restante (posición 12). Por último, vamos a ver las rutinas necesarias para leer uno o más bloques en memoria, asimismo como el de la grabación. Para ello hay que utilizar estructuras muy específicas, pero el resultado final consiste en un programa de muy fácil uso y de corto espacio. Primero veamos la rutina para la lectura del bloque 880 (ver listado 1).

El programa utiliza la estructura comentada anteriormente, IORequest, encargada de efectuar las funciones primarias de lectura, grabación, etc., de bloques. Observa cómo maneja en la parte del PROGRAMA esta estructura, insertando en los offsets $1C,$2C,$24 y $28 los valores respectivos de comando, offset del bloque, longitud y posición de memoria del buffer para la lectura o para la grabación. En cuanto a los comandos en offset $1C xisten los siguientes:

1 RESET
2 LECTURA
3 GRABACION
4 VACIADO DE BUFFER
5 BORRADO DE BUFFER
6 PARADA
7 COMIENZO después de la parada
8 ABORTAR proceso parada-comienzo
9 NO ESTANDAR

Los que se utilizan generalmente son el 2, 3, 4 y 9.

En cuanto al offset del bloque hay que tener en cuenta una fórmula: 512 x (S+11 x H+11 x 2 x C) en la que S es el sector, H es la cabeza y C el cilindro. Para el caso del bloque 880 (directorio principal) el sector es el 0, la cabeza 0 y el cilindro el 40. La operación nos da el valor 450560 (hexadecimal = $6E000), para luego insertarlo en el offset correspondiente. La longitud es en realidad el número de bloques a leer a partir del bloque indicado. Para un bloque son 512 bytes (= $200), dos bloques 1024 (= $400), hasta realizar una multiplicación sencilla de 512 por el número de bloques. Lo último que queda es la posición de memoria en la que se hara el proceso. En nuestro caso no hemos tenido espacio para insertar una rutina de reservar memoria, utilizando por tanto una zona un poco alta ($60000), pero recomendamos reservarla en CHIP MEMORY pues el acceso a bloques se realiza en este rango. Ahora que ya sabemos todos los parámetros para la lectura del bloque basta simplemente ejecutar la función DoIO en el offset -$1C8 de la librería EXEC. También hemos utilizado comandos adicionales (UPDATE y NO ESTANDARD) para dar paro al motor del drive. Para realizar una grabación de un bloque el proceso es exactamente el mismo, con la única diferencia del comando, en este caso el 3, y tener preparada la zona de memoria que va a ser llevada a disco. Nada más por el momento. Espero que este capítulo le haya sacado de algunas dudas que tenía hace tiempo. ■

ERROR DE CARGA

Hace unos días recibí el disco especial Amiga World 3, y tengo un problema con él que no sé si es debido al poco tiempo que tengo el ordenador o al mal funcionamiento del disco.

El problema es el siguiente:

Cargando música......

CLI 3

Después de esto me sale un «Requester» que dice lo siguiente:

Volume
Amiga World 3
has a read/write error

Retry Cancel

Y de aquí no puedo pasar.

J. Antonio Redondo Sánchez
Valencia

Dicho error se produce debido a que el disco, al ser introducido, no debe estar protegido contra escritura. Ello se debe al mecanismo usado para la carga de la música que el disco contiene, por lo que la solución es simple:

Al introducir el disco asegúrese de que no está protegido contra escritura.

QUEJAS DEL LECTOR

Acabo de recibir el número cinco de Amiga World y la impresión que me ha causado es de estupefacción: ¿a qué viene incluir catorce páginas dedicadas al C-64 (casi un 20% del total de la revista?, ¿no tienen los usuarios de este ordenador una revista especial para ellos?, ¿acaso podemos, los usuarios de Amiga sacar alguna utilidad a estas páginas?

Supongo que el problema que tienen ustedes es que no tienen suficiente material para incluir en la revista, porque sino tampoco se entiende que empiecen una nueva sección de lenguaje ensamblador por la séptima parte.

Desearía seguir disfrutando de una revista seria para un ordenador con el que aparte de jugar creo que se puede utilizar para hacer cosas serias, y mucho me temo que o me he equivocado de ordenador, o me he equivocado de revista.

Mora i Ayza, Jordi
Llavaneres (Barcelona).

1. En principio la nueva revista Amiga World es una continuación de la anterior publicación para usuarios de ordenadores Commodore (anteriormente usuarios del modelo C-64).

La inclusión de las catorce páginas mencionadas es debido a que efectivamente los usuarios de C-64 ya no tienen una revista exclusiva para ellos, igual que en un principio eran los usuarios de Amiga los que no tenían una revista propia, por lo que se incluían páginas para el Amiga dentro del Commodore World (caso que ocurrió anteriormente entre VIC-20 y C-64).

2. Partiendo de la base de que Amiga World es una prolongación de Commodore World (al igual que la mayoría de los lectores cambiaron su ordenador C-64 por un nuevo Amiga), puede comprender que no se hayan incluido los seis primeros capítulos del lenguaje ensamblador dentro de Amiga World, puesto que dichos artículos se incluyeron anteriormente en Commodore World.

Es por tanto que no sería de gran utilidad (ni grato), para una gran parte de los lectores, encontrarse de nuevo con una serie de capítulos referidos al lenguaje ensamblador los cuales ya fueron publicados anteriormente.

AMIGA Y LA MUSICA

Soy un usuario de Amiga al que le entusiasman las posibilidades sonoras de este ordenador. Deseo utilizar para mis músicas instrumentos «Sampleados» como los que se escuchan en muchas melodías de Video juegos, pero no sé cómo he de hacerlo.

Ernesto García
Orense.

Efectivamente las posibilidades del Amiga destacan en lo que se refiere a la música, no en vano es un ordenador utilizado por músicos de gran prestigio dentro de sus composiciones.

Para poder sacar un buen rendimiento a este ordenador y no limitarse a los instrumentos que pueda incluir un editor, se deben poseer los siguientes medios:



— **Un digitalizador de sonido (por ejemplo Future Sound).**
— **Un buen programa sampleador (Audi Master II).**
— **Un programa editor (Sonix).**

Es importante que la fuente de digitalización sea de la mayor calidad posible, así como disponer de mucha paciencia para los ajustes de frecuencia. Asi mismo la duración de un sampleado dependerá de la cantidad de memoria disponible.

Para una mayor información sobre este tema puede leer los artículos incluidos en las revistas cinco y seis de Amiga World, en las cuales se explica todo lo referente al uso de digitalizadores, creación de instrumentos y consejos a la hora de digitalizar.

PROBLEMA DE ALMACENAJE

Hace poco tiempo que poseo este maravilloso ordenador, el cual me compré con fines serios. El problema con el que me encuentro, debido seguramente a mi inexperiencia, es el siguiente:

Cuando me encuentro usando una aplicación, y me dispongo a grabar a disco el resultado, resulta que el disco de trabajo está lleno y no dispongo de ningún otro disco formateado; con lo cual se pierde todo el trabajo. A todo ello y si tuviera algo que ver con el mencionado asunto, la configuración de mi ordenador es la siguiente: Amiga-500, 1 unidad 3.50, 1 mega de memoria.

Alejandro Linares. Vigo.

Decir que siempre es conveniente tener algún disco formateado en el caso de que se presenten «inconvenientes», ya que es muy decepcionante ver cómo tras vario tiempo de trabajo, damos con éste en la basura.

Una posible solución, la cual depende de la aplicación usada (esta no se indica en su carta) es la siguiente:

Poner la ventana de la aplicación en segundo plano, introducirse en el CLI, formatear un nuevo disco, Salir del CLI, poner la ventana de aplicación en primer término y por último seguir trabajando.

Si la ventana de la aplicación no tiene control para decidir el plano (fondo o primer término), no habría otra solución que la de «resetear» el ordenador y empezar de nuevo. ■

Esta sección es un buzón abierto a todo tipo de sugerencias, comentarios y consultas de los lectores. Envía tus cartas a: Commodore World, Rafael Calvo,18, 4.º B. 28010 Madrid,

WAR IN MIDDLE EARTH

A quién le gusten los libros de aventuras, seguro que se ha leído «El señor de los anillos». Este libro y su antecesor «El Hobbit», son quizá los que marcaron la pauta de todo este tipo de libros. «War in Middle Earth», refleja fielmente todos los datos que éstos libros nos dán de personajes como Frodo y Gandalf el mago.

La finalidad del juego, es que el Señor Oscuro de Mordor, no consiga el anillo que él mismo hizo para gobernar a todos los habitantes de la Tierra Media. La ambición del Señor Oscuro, es cada día mayor y si no se hace algo, acabará por invadirlo todo. La guerra parece inevitable, pero no será ganada por Sauron (El Señor Oscuro) sin la ayuda del anillo.

La acción se desarrolla de distintas formas, tanto en pantalla de información conversacional, como visualizando a Frodo y sus amigos en su marcha hacia Mordor para intentar destruir el anillo. Durante el juego es conveniente estar contínuamente informado de dónde se encuentran los enemigos. Esto se consigue visualizando continuadamente el mapa de situación, en él, se ve tanto nuestra posición como de todo el entorno.

Es una pena, que aunque las instrucciones son en castellano, la información que ofrece el programa através de la pantalla es en ingles, y cuando el tiempo apremia, la dificultad al traducir se convierte en un nuevo enemigo. El juego es realmente difícil y requiere una gran dedicación. Pero si se ha leído el libro y le gustan los juegos de estrategia, éste es de lo mejor.

Las instrucciones que se proporcionan con el programa son importantes ya que hacen un resumen del libro en que se basa el juego, y nos dan una relación de los participantes en él. En la relación se indica la importancia y si es amigo o no de nuestra causa. Así mismo se indican las coordenadas de los lugares que van apareciendo.

La presentación del programa es buena pero lo realmente destacable, es que los dos discos de que está compuesto el juego, están llenos de pantallas... y pantallas, tantas como un país. Los personajes que aparecen no están muy reconocibles pero se salva con la cantidad de ellos existente. Solo a Frodo, en un principio le acompañan dos Hobbits pero luego se van uniendo, un Elfo etc. hasta llegar a ocho acompañantes.







El acoso de las fuerzas del mal puede desembocar en una lucha sangrienta en la que se puede elegir dirigirla o participar en la batalla con algún personaje. Para las luchas hay que contar con los aliados enanos etc. para poder ganar. Es de destacar, que los más feroces y habituales enemigos, los Orcos, temen a la luz aunque no llega a ser del todo su enemiga. Pero sí que se desenvuelven mejor durante la noche.

Además de los habitantes de las tierras de Mordor, que quieren el anillo para vencer y establecer el mal. Está el Sumo Mago Saruman que, influenciado por el mal, quiere para sí el anillo y con él poder dominar la tierra. Saruman, no se da cuenta de que cualquiera que domine la tierra gracias a ese anillo, poco a poco se irá convirtiendo en un nuevo Señor Oscuro debido a la maldad para que se creo el anillo.

En resumen, War in Middle Earth, es un buen juego de estrategia y acción en el que hay que estar informado de la historia y los personajes, así como dedicarle bastante tiempo para poder «empezar» a sacarle partido.

Como ya indicamos, el juego para ser acabado necesita mucho tiempo... casi tanto como el libro. El único defecto que es posible encontrarle, es lo lento que puede hacerse en determinadas partes, sobre todo al principio, y también un poco los gráficos. En ellos falta más claridad y definición, para mi gusto sobran demasiados puntos que además, no parecen estar en su sitio. Los colores de las ciudades así como de ríos etc., no me parece que estén muy bien elegidos. Por ejemplo en los campos el color verde tiene esparcidos demasiados puntos de otros colores, que en vez de originalidad, dá la sensación de que está mal dibujado. ■

THE GAMES SUMMER EDITION

En el mundillo de los videojuegos hay o simplemente juegos o verdaderas obras de arte, en este caso The Games Summer Edition se corresponde con el segundo apartado. Es sorprendente ver cómo de vez en cuando aparecen juegos en el mercado que merecen realmente la pena, ya que absolutamente todos los detalles así como los efectos del juego se encuentran perfectamente cuidados.

Como ya habrán adivinado, The Games Summer Edition es un juego de carácter deportivo de la casa EPYX. Este cuenta con diferentes pruebas en las que, como anteriores juegos deportivos de esta misma casa, se podrá o bien practicar o competir.

La presentación del juego es verdaderamente buena con animación tridimensional. Una vez dentro del propio juego se presenta la pantalla de selección de pruebas en forma de iconos, así como otro icono especial para acceder a la pantalla de menú.

La pantalla de menú resulta bastante completa, las opciones que ésta presenta son comparables a las de otros juegos deportivos de esta misma casa como pueda ser el Summer games. Las diferentes opciones son las siguientes:

Empezar competición: con esta opción tendremos que seleccionar los nombres y países de los diferentes competidores, así como las pruebas en las que se desea competir.

Practicar pueba: esta opción permite practicar a un número de participantes en las diferentes pruebas.

Ver récords: con esta opción se visualizará por pantalla los diferentes récords de las pruebas.

Inicializar récords: con esta opción se podrán inicializar los récords; esta opción es muy útil para aquellos participantes que sin poseer muy buenas marcas deseen ver su nombre inscrito en dicha lista.

Volver a la presentación: seleccionando dicha opción podrá volver a repetir la vistosa presentación del juego.

Volver a la pantalla de selección de pruebas: con esta opción se volverá a la pantalla donde se seleccionarán las diferentes pruebas, ya sea para competir o practicar.

Las diferentes pruebas que contiene el programa son las siguientes: ciclismo, 400 vallas, martillo, arco, barras asimétricas, anillas, salto con pértiga, salto de trampolín. Dichas pruebas se encuentran divididas entre los dos discos de los que se compone el juego, contando todas ellas con efectos muy, muy cuidados y simpáticos.

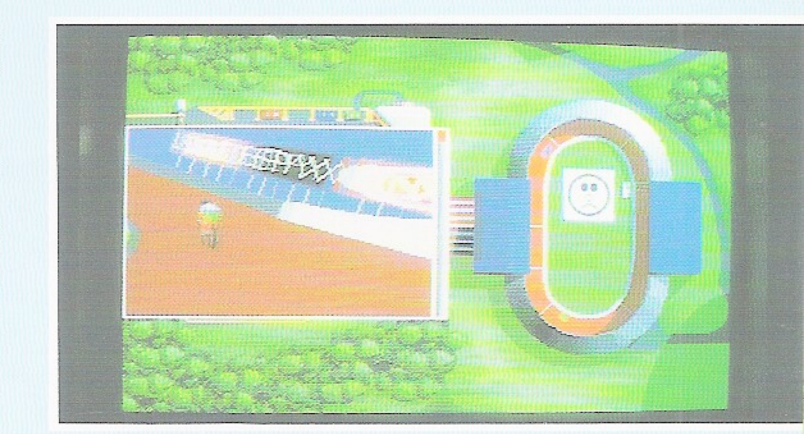
Los gráficos de todas las pruebas son espectaculares, así como las perspectivas de algunas de ellas las cuales son en tres dimensiones. En el apartado del sonido decir que se aprovecha esta cualidad a fondo, ya que como en el resto de los apartados está perfectamente cuidado.

El juego se completa con unas completas instrucciones sobre el manejo del mismo. Lo único que se podría comentar en el apartado negativo de este juego es la constante necesidad de cambiar de disco, lo cual no se hace especialmente pesado puesto que merece realmente la pena.

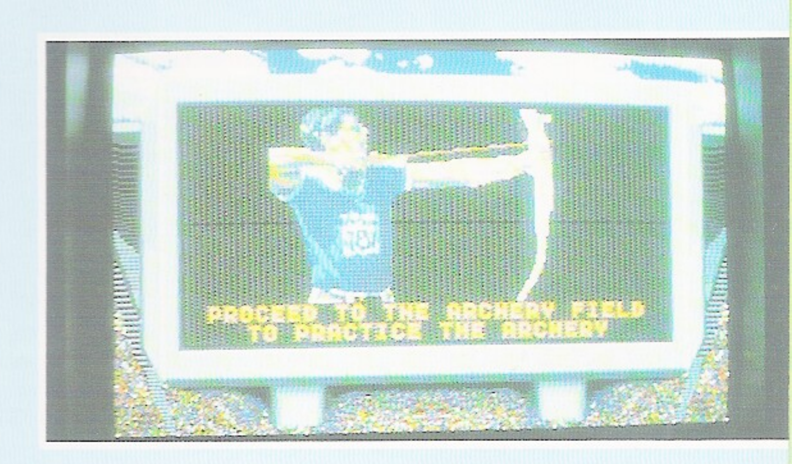
Por último decir que, cuando la compañía Epyx edita un videojuego, siempre lo hace bien. Y el resultado está a la vista...

Un programa lleno de efectos curiosos y bien cuidados, de músicas de acompañamiento que agrada escuchar en todo momento. Será muy útil que las productoras, a la hora de comercializar un video juego, mirasen programas como este. ■











MISSION ELEVATOR

Trevor, a quién Ud. interpreta cuando juega, es un superagente del FBI. Le han encomendado una peligrosa misión en un hotel, tapadera de grandes secretos del FBI. En el hotel, unos espías extranjeros tienen puesta una bomba que amenaza con destruirlo. Se tiene información de que la bomba está en el piso 62. Hay que llegar a ese piso y desactivarla. Siempre y cuando se logre esquivar a la gran cantidad de espías que llenan el hotel y que aparecen por todas partes.

Este juego, ya fue interesante en la versión que se realizó para el C-64 y también, es conocido por su divulgación en las máquinas recreativas. La versión realizada para el Amiga, podía haber sido más acertada, sobre todo en el movimiento y sonido, todos sabemos que con el amiga es posible mucha más espectacularidad.

Mission Elevator es un juego que crea adición, siempre que le gusten este tipo. La infinidad de pantallas(son en total 64 plantas con sus respectivos ascensores, habitaciones, muebles etc.), hacen que el juego sea realmente de adictos ya que sino, es difícil lograr el objetivo. La utilización de objetos como llaves para conseguir



entrar en habitaciones y también para poder subir a otras plantas, logran acabar con nuestra paciencia en varias ocasiones.

Los movimientos de Trevor, como el salto y el agacharse están conseguidos y son rápidos. los sonidos de los disparos así como la continua aparición de los mismos enemigos (espías extranjeros) hacen que la generalidad del juego, nos parezca bastante parecido. A ésta igualdad contribuye mucho la cantidad de ascensores.

La presentación del programa deja mucho que desear y en general se podría haber hecho un juego mucho más espectacular, pero sin ser un juego con grandes pretensiones, entretiene durante... un buen rato.

Con respecto a la ya conocida versión de Commodore 64, en el Amiga están mucho mejor los gráficos, como es normal, y también la velocidad así como que es más extenso. ■





BEACH VOLLEY

Los videojuegos deportivos siempre han estado de moda, esta vez nos encontramos con uno más de la mano de Ocean: el Beach Volley.

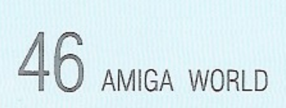
Este es un juego sorprendente desde la carga del mismo, ya que los gráficos de presentación, así como la forma de realizar la misma son tremendamente espectaculares.

Una vez finalizada la carga del juego, se representa un «sketch» que nos introduce en la acción; a partir de este punto ya sólo quedará elegir entre las diferentes opciones que se presentan para iniciar el juego.

Dentro de las diferentes opciones que presenta este juego hay una especialmente interesante: la de juego a dos jugadores. Dentro de esta opción nuestro contrincante será otro jugador en vez de estar manejado por el ordenador.

Las reglas del juego son en todo momento las del juego del Volley Ball, disponiendo de todas las acciones posibles que este juego posee. De esta forma se tendrán varias posibilidades de saque, mate, o simplemente toque.

Los gráficos de los personajes así como sus animaciones son tremendamente realistas. En general este es un juego en el que pasar mucho tiempo de ocio no resultará difícil, ya que el cuidado puesto dentro del juego consigue que éste nos atraiga en todo momento. En el aspecto sonoro Beach Volley se encuentra lleno de buenos efectos sonoros, así como una bonita melodía; el resultado es un juego que aunque no innovador, sí resulta grato tenerlo en la videoteca del Amiga. ■





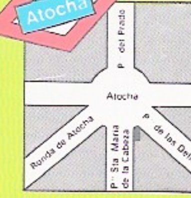
VENTA POR CORREO

HORARIO DE PEDIDOS

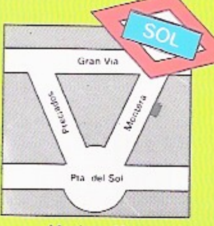
LUNES A VIERNES 10,30 A 20,30	SABADOS DE 10,30 A 14,00	(91) 239 34 24 / 239 04 75



TIENDAS
MAIL SOFT



Pº Sta. María de la Cabeza, 1
28045 MADRID
Telf. 227 82 25



Montera, 32
Telf. 522 49 79

Ordenadores

AMIGA 500 — 94.900 ptas.

AMIGA 2000 — 215.000 ptas.

Manuales en castellano.
Garantía oficial Commodore España

OFERTAS

Ampliación ½ Mb con reloj: 23.900
Unidad de disco 3 ½ externa: 22.900
Impresora STAR
LC 10 color: 49.900
LC 10 b/n: 39.900

Periféricos

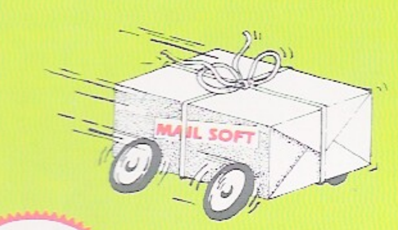
	PRECIO
Monitor 1084	55.000
Cable Euroconector	3.500
Filtro Especial 1084	4.500
Unidad de 3 1/2 con lector de pistas	27.500
Unidad de 5 1/4 con lector de pistas	34.000
Unidad disco interna A2000	18.900
Ratón	8.900
Digitalizador de Sonido	13.900
Digiview Gold V.4 (novedad)	28.000
Modem externo 2400 baudios	26.600
Senlock NHS	49.900
Separador de colores NHS	34.900

Utilidades

PROGRAMA	PRECIO	PROGRAMA	PRECIO
Pagesetter	15.900	Video Effects 3D (Pal)	33.900
Pagestream	28.900	Video Page	21.750
Pagestream Font	5.800	Video Tittler Aegis	21.900
Professional Page	49.900	Videoscape 3-D	28.900
Data Retrieve	10.900	X-Cad Designer	21.750
Superbase Personal	11.900	Zoetrope	19.900
Superbase Personal II	19.900	Maxi-Plan 500	19.900
Diga	11.500	Maxi-Plan Plus	29.900
On line Platinum	14.500	Assem Pro	14.500
Comic Setter	13.900	Aic Fortran	42.900
Deluxe Paint II Pal	12.500	Devpac	14.500
Deluxe Paint III	20.900	Hi Soft Basic Pro	22.900
Deluxe Photolab	12.500	Multi Forth	14.500
Deluxe Print II	11.600	Lattice C. Ver. 5.0	43.500
Deluxe Video	12.500	Aegis Sonix 2.0	10.900
Digi Paint 3	14.500	Audiomaster II	13.900
Express Paint	17.900	Deluxe Music	12.500
Fantavision	7.900	Dynamic Drums	9.500
Intro Car	9.900	Midi Rec. Studio	9.900
Movie Setter	14.500	Music Studio 2.0	11.600
Photon Pain 2.0	19.900	Kind Words (español)	12.500
Printmaster Plus	5.800	Profont (1,2)	6.900
Sculp 3D	14.500	Prowrite	17.000
TV Show	13.900	Wordperfect	45.000
TV Text	13.900	Zuma Fonts (vol. 1,2,3,4)	6.900

NOVEDADES

Ampliación A500 Ampliable 2Mb
1/2 Mb - 32.300
1 Mb - 39.000
1/5 Mb - 45.800
2 Mb - 52.500
Unidad disco 3 1/2
1011 Commodore - 36.000



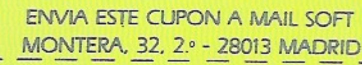
JUEGOS AMIGA

JUEGOS AMIGA	
2000 L. Viaje Submari (N)	2.850
Altered Beast	1.990
Baal	2.250
Barbarian II	2.500
Beach Volley	1.990
Blood Money	2.250
Bomber (N)	2.850
Cazafantasmas 2	2.850
Chase H.Q. (N)	2.250
Chessplayer 2150	3.995
Continental Circus (N)	2.500
Demon Stalker (N	2.500
Double Dragon	2.500
Dr. Doom's Revenge	2.250
F-16	4.995
Falcon (N)	5.000
Football Manager II	2.450
Galaxy Force (N)	1.990
Gazza's Super Soccer (N)	2.250
Gemini Wing (N)	2.500
Ghost's & Gouls	2.250
Hard Driving (N)	2.250
Hermes (sim. vuelo)	4.995
Indiana Jones	2.250
Jabato	2.500
Kick Off	2.850
Knight Force	2.250
Liverpool (N)	2.250
Moonwalker (N)	2.250
New Zeland Storys	1.990
North and South	2.250
Oil Imperium (N)	2.250
Paris Dakar (N)	2.850
Pictionary	2.750
Pimball Magic	2.250
Pro Tennis Tour (N)	3.900
Power Drift	1.990
Rick Dangerous	1.990
Robocop	1.990
Rock'n'Roll	2.250
Shinobi	2.500
Silk Work	1.990
Space Harrier II (N)	2.250
Super Wonder Boy	1.990
The Shadow of Beast	2.250
Tin Tin (N)	2.250
Trivial Pursuit	2.750
Turbo Out Run (N)	2.250
Tusker (N)	2.250
War in the Middle E.	2.500
Wild Street (N)	2.250
Xenon 2	2.250

SI TU PEDIDO ES SUPERIOR A 20.000 PTAS. TE LO ENVIAMOS A CASA EN MENOS DE 24 H. Y SIN GASTOS DE ENVIO.



Todos nuestros precios incluyen I.V.A.



ENVIA ESTE CUPON A MAIL SOFT
MONTERA, 32, 2.º - 28013 MADRID

NOMBRE
APELLIDOS
DIRECCION COMPLETA
POBLACION ____ PROVINCIA
TELEFONO ____ C.P.
MODELO ORDENADOR
N.º CLIENTE
☐ NUEVO CLIENTE
FORMA DE PAGO:
CONTRA REEMBOLSO

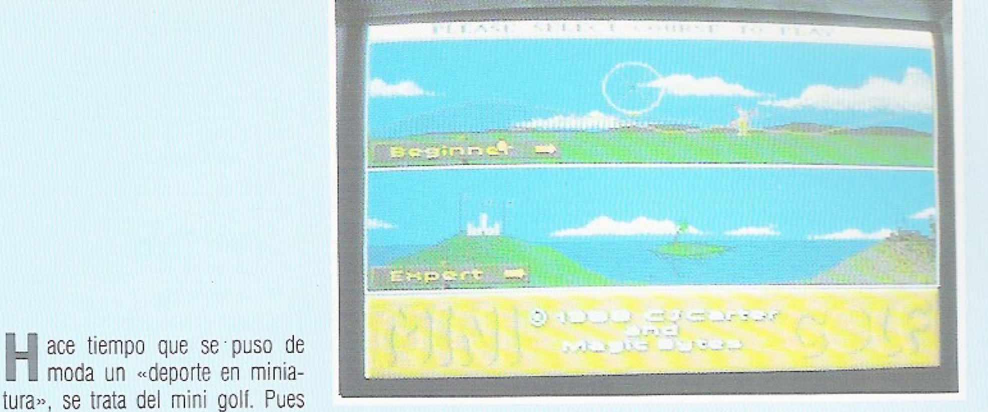
TITULOS DE PEDIDOS	PRECIO
☐ LISTADO DE PROGRAMAS	0
GASTOS DE ENVIO	200
TOTAL	

REVISTA A-TOOL

Revista AMIGA TOOL (en español)
Dos discos llenos de trucos para juegos, información, noticias y software de dominio público por sólo 995 ptas.

EN NUESTRO CATALOGO PUEDES ENCONTRAR MUCHO MAS. SOLICITALO

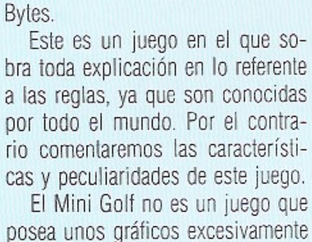
10 DISCOS VIRGENES
1.490



MINI GOLF

Hace tiempo que se puso de moda un «deporte en miniatura», se trata del mini golf. Pues bien como es costumbre en todo lo que se pone de moda ahora podrá disfrutar de la versión informática para Amiga de este juego de la mano de la conocida casa Magic Bytes.

Este es un juego en el que sobra toda explicación en lo referente a las reglas, ya que son conocidas por todo el mundo. Por el contrario comentaremos las características y peculiaridades de este juego.



El Mini Golf no es un juego que posea unos gráficos excesivamente espectaculares, sino que éstos son simplemente prácticos, lo cual no quiere decir que en algunos momentos del juego no sean graciosos o que éstos no estén bien realizados.

En el Mini Golf a diferencia del golf grande no se posee más que un tipo de palo para golpear la bola, por lo que no existen las indecisiones a la hora de elegir el tipo de palo.

La mecánica del juego es la siguiente: en un principio se deberá elegir el recorrido en el que se desea jugar, existiendo dos posibles (beginer, expert). Una vez elegido el recorrido, aparecerá el primer hoyo de éste, en el cual el put que se debe hacer en dicho hoyo se verá representado en la parte superior de la pantalla.

Si a lo largo del juego los golpes realizados se pasan del put indicado en cada uno de los hoyos, el ordenador pasará automáticamente al siguiente de los hoyos, sumando cinco puntos de penalización.

En el apartado sonoro decir que éste se limita al «poc» de turno, es decir un ruidillo cada vez que se golpea la bola. En general el Mini Golf es un juego que cumple su cometido: entretener. En este juego los gráficos superespectaculares seguramente no tienen razón de ser, la música o melodía no existe, ya que lo único que importa en este juego es meter la bola en el hoyo. ■

DRAGON SPIRIT

Seguramente ya no sorprenderá a nadie ver en las pantallas de su ordenador juegos en los que el único objetivo es el de aniquilar todo lo que aparezca en la pantalla.

Dragon Spirit es uno de esos juegos arcade en los que el único objetivo es destruir lo que se encuentra a su paso, pero la falta de un argumento innovador se ve compensada por otros aspectos.

Lo que realmente «salva» a este juego es, sin lugar a dudas, su enorme grado de adicción ya que es bastante difícil «despegarse» de la pantalla del ordenador una vez que se ha empezado a jugar.



Si bien la dificultad del juego es excesiva a medida que se avanza en el desarrollo del juego, se puede decir que no por ello decae el interés del jugador ya que la gran velocidad a la que se desarrollan todas las acciones hace que éste no sea «pesado».

En el apartado gráfico decir que este juego se encuentra entre la media general, ya que los diferentes fondos así como los personajes del juego no destacan espectacularmente, pero sí cumplen perfectamente su cometido.

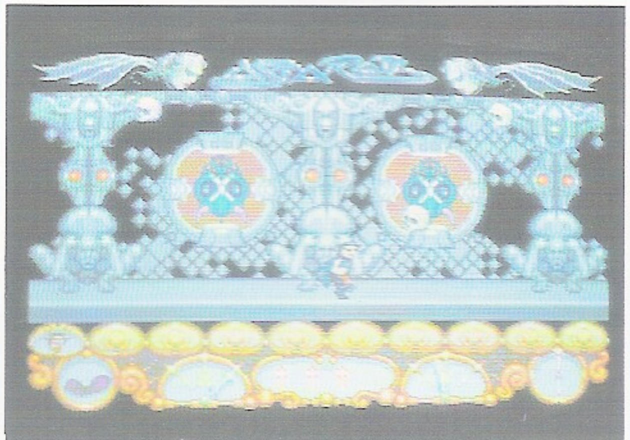


El juego en sí nos muestra como personaje protagonista a un dragón volador, el cual podrá aumentar su poder a medida que dispare a unas criaturas brillantes y cogiendo los hechizos que éstas sueltan. Los diferentes extras pueden variar desde más cabezas para el dragón, hasta fuerza de disparo.

Decir también que como objetivo final, y después de haber completado los nueve niveles de los que se compone el juego, el protagonista podrá rescatar a la princesa Alicia, la cual ha sido secuestrada por la serpiente Zawell. Por tanto será necesario armarse de paciencia y no desesperarse fácilmente.

En definitiva se puede decir que este juego es imprescindible en la biblioteca de los MUY MANIACOS de este tipo de juegos, ya que reúne todas las características necesarias: gran grado de adicción, dificultad y muchos niveles de juego; por lo que completar el mismo será un gran reto. ■





ASTAROTH

Entre la avalancha de juegos que salen cada mes para Amiga, es difícil ver algo que sea diferente a lo demás; en este caso Astaroth sigue esta norma, ya que no es un juego que se pudiese llamar innovador.

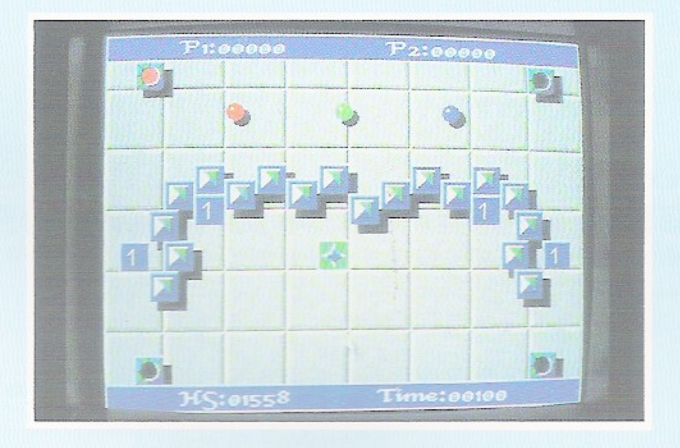
El argumento de Astaroth es el siguiente: el protagonista del mismo (Ozymandias) deberá viajar a través del reino de las catacumbas, en busca de los nueve poderes mentales que se hayan distribuidos por las diferentes estancias. Estos serán imprescindibles si se quiere completar el juego, ya que servirán a nuestro protagonista para vencer a la malvada Astaroth.

Los diferentes poderes mentales que deberá reunir Ozymandias son los siguientes: Telequinesia, con la cual podrá desplazar objetos; piroquinesia, con esta aptitud podrá crear fuego (disparar fuego) con lo cual será muy útil para destruir a varios enemigos; telepatía, con este poder podrá leer el pensamiento de los demás personajes, diciendo además adónde deberá dirigirse el personaje; cambio de forma, una vez Ozymandias posea este poder, podrá convertirse por ejemplo en pájaro; transmigración, con este poder el personaje se podrá desplazar de un lugar a otro sin necesidad de caminar; crioquinesia, este poder sirve para congelar objetos con la mente; visión nocturna, este es el último de los poderes mentales que podrá tener Ozymandias; dicho poder le será muy útil para poder ver en las estancias oscuras.

Sin embargo la obtención de la totalidad de los poderes será sin lugar a dudas una ardua tarea para el jugador, ya que éstos se encuentran bastante dispersados por entre las calacumbas. Asimismo para poder encontrar alguno de los poderes será necesario estar en posesión de otro de éstos, ya que de lo contrario resultaría imposible la obtención de los mismos.

El nivel de dificultad del juego, así como el de adicción del mismo son bastantes elevados; pudiendo llegar al final del mismo tras muchas horas de juego y dominio de cada uno de los diferentes poderes. En el aspecto gráfico reseñar la calidad de los mismos que, aunque poco variados, son bastante buenos; en el aspecto sonoro decir que no es ninguna maravilla pero consigue su propósito.

En definitiva Astaroth es un juego curioso. ■



BEAM

«Los ojos del jugador-maestro exploran el primer nivel, él es un muelle tensado, un gato salvaje listo para atacar. Como un rayo, desde el cielo descubre las soluciones y entiende las conexiones que se establecen entre los objetos que ve. Este raro paisaje no le es desconocido, parece como si ya hubiera estado aquí antes».

Según la carátula este párrafo que han leído corresponde a la descripción del juego, pero como de costumbre los guiones sólo se quedan en «derroche» de imaginación, teniendo algo de sentido solamente la frase final. En realidad de lo que se trata en el juego es de «haz algo en cada pantalla y con cuidado que sólo tienes una vida de principio a fin...».

En realidad el único aliciente que posee este juego es el reto de acabar unas cuantas fases antes de arrinconar el mismo, ya que éste no resulta atractivo de jugar en ninguno de los casos.

Los gráficos de Beam son simples y pobres, ya que con unos cuantos gráficos se resuelve la construcción de las diferentes pantallas de juego. El movimiento de nuestra ¿nave? no resulta complicado, pero sí que resultará difícil llevarla en el sentido deseado y por último el juego es, en conjunto, aburrido.

Entre las diferentes opciones de menú que presenta Beam se encuentra la de uno o dos jugadores, así como el tipo de nave que se desea pilotar (planeadora o misil).

La cantidad de puntos finales que se posea será la suma de los obtenidos en cada fase, estando éstos en proporción al tiempo usado en completar cada fase; por lo que si se desea conseguir un buen «score», será necesario hacerlo lo más rápido posible.

Es indudable que el Amiga es una de las mejores máquinas de juegos, lo cual no quiere decir que no sea también una buena máquina en otros aspectos de la informática, pero si se desea convencer a alguien de esta cualidad es preferible no hacerlo mostrándole este juego.

En definitiva decir que Beam es un juego que pasará seguramente sin pena ni gloria por los anales de los videojuegos, ya que la calidad de un juego no puede ser sustituida por un argumento en el que se derrocha imaginación, y a veces demasiado... ■

COMMODORE AMIGA WORLD



Desde enero de 1990 AMIGA WORLD tendrá periodicidad mensual.
SUSCRIBASE AHORA Y PAGUE EN ENERO DEL 90

VENTAJAS QUE OBTIENE AL SUSCRIBIRSE AHORA

AHORRO de 825 pts. 15 % de descuento sobre el precio de portada, 11 ejemplares. Usted sólo paga 4.675 pts.

GRATIS

el ejemplar de AMIGA WORLD correspondiente a noviembre 89

BOLETIN DE SUSCRIPCION AMIGA WORLD

Para suscribirse ahora o solicitar más información, sólo es necesario enviar este boletín a: CW Communications (Amiga World), Rafael Calvo, 18, 4.° B, 28010 Madrid.

Nombre .... Empresa ....

Dirección .... Población ....

Provincia .... C.P. .... Teléfono ....

Modelo de ordenador y configuración ....

Marque los recuadros con una «X».

- ☐ Suscripción ahora, pagando en enero de 1990 .... 4.675
- ☐ Pago anticipado ahora (ahorro de un 10 % adicional) 4.125
- ☐ Suscripción revista + disco (11 revistas + 11 discos) 18.000
- ☐ Deseo recibir más información sobre Amiga World.

Forma de pago: Cheque, giro o tarjeta de crédito. FIRMA ....

☐ Cheque ☐ Giro (indicar n.°) .... importe ....

☐ VISA ☐ MASTERCARD Número de tarjeta: ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

FECHA DE CADUCIDAD ....

Enviar a: CW COMMUNICATIONS AMIGA WORLD. Rafael Calvo, 18, 4B. 28010-MADRID

IMPORTANTE: Además de las ventajas que le supone suscribirse ahora a la revista AMIGA WORLD, si prefiere hacernos efectivo el importe por anticipado, tendrá un descuento adicional de un 10 %, con lo que el precio total de la suscripción sería de 4.125 pts., un ahorro de 1.375 pts.



Para realizar sus pedidos copie o fotocopie esta página completa, marcando lo que desee con una cruz e indicando todos sus datos. No se atenderán pedidos con datos incompletos. Sume usted mismo el importe total del pedido y envíenos un cheque, giro o los datos de su tarjeta de crédito. Si paga por giro, indique también su pedido en el texto del resguardo.

REVISTAS AMIGA WORLD

0 | 1 | 2 | 3 | 4 | 5 | 6

500 pesetas cada ejemplar

DISCOS AMIGA WORLD

0 | 1 | 2 | 3 | 4 | 5 | 6 | 7

1.700 pesetas cada unidad

DISCOS AMIGA DE LA REVISTA COMMODORE WORLD

1 | 2 | 3 | 4 | 5 | 6

1.700 pesetas cada unidad

OFERTA REVISTA MAS DISCO: 1.995 ptas.

- Tres discos .... 4.500 ptas.
- Cinco discos .... 7.000 ptas.
- Once discos .... 14.000 ptas.

DISCOS AMIGA DE LA REVISTA COMMODORE WORLD

- **Disco Amiga 1.** (Correspondiente a las revistas 45, 46 y 47 de Commodore World.) Contiene: Biorritmos, Checksum 1.0, Filedit, InvaderCraft, Listador.
- **Disco Amiga 2.** (48, 49 y 50). BeatMaster, Fractal Maker, IFF. Transfer, IFF. View, Music, Ray-tracing, Reversi, Listador.
- **Disco Amiga 3.** (51, 52, 53 y 54). Curso de C (I al IV). Listador, Music Machine III, VideoText.
- **Disco Amiga 4.** (55, 56 y 57). Curso de C (V al VII). Superb! y el juego de regalo Sidewinder.
- **Disco Amiga 5.** (58, 59 y 60). Curso de Ensamblador 1 y 2. Checksum 2.1. Programa de regalo: GADGETS BASIC.
- **Disco Amiga 6.** (61, 62, 63 y 64). Curso de Ensamblador 3, 4, 5 y 6. Programas de Intros para tus discos como regalo.

Nombre y apellidos ....

Dirección ....

Población .... C.P. .... Provincia ....

Teléfono .... Modelo de Ordenador ....

Importe del pedido .... ptas.

Forma de pago: ☐ **Cheque** ☐ Giro número ....

☐ VISA ☐ MasterCard N.º ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

Fecha caducidad ☐☐ ☐☐

Firma

Los pedidos con **tarjeta de crédito,** sólo a partir de **3.000 ptas.**

Gastos de envío e **IVA incluidos.**

NO SE ADMITEN **PEDIDOS CONTRA REEMBOLSO**

MINIGEN

Fabricante: Progressive Peripherals & Software.
Distribuidor: Ibergen.
P. Verdún, 33-39,kl esc izq. 6°, 1ª
08031 Barcelona
Tel.: (93) 353 19 73.

Progressive Peripherals í Software, creadores de ProGen, ofrecen ahora otro nuevo genlock para el Amiga.

MiniGen, una pequeña caja de 6.25 x 1.50 x 2.25 pulgadas, posee únicamente tres puntos de conexión. Este Genlock posee en su parte final dos conectores hembra para la conexión de cualquier tipo de fuente externa (cámara, video, u otro tipo de fuente) y crear una señal de video compuesto paar VCR o monitor. Sobre el otro conector, un conector hembra de 23-pin D, conecta directamente el puerto RGB del Amiga al Genlock. Esta configuración elimina la necesidad de cables, los cuales podrían crear problemas en algunos casos; un display de selección permite asimismo la elección de señal de video, señal de gráficos, o las dos señales mezcladas.

Este Genlock, aunque no de prestaciones profesionales, sí presenta unas buenas prestaciones comparando precio/calidad. Así mismo las instrucciones del mismo se encuentran totalmente en castellano. Es debido a ello por lo que el uso de este Genlock se simplifica en extremo.

Con una fuente externa de señal, la salida video de MInigen es poco efectiva; pudiendo conectar una cámara, VCR. Esta configuración acortará la vida del tubo en un tubo tipo video cámara, excluyendo esto, este paquete no tiene ningún otro punto en contra.

Se ha probado el Minigen con las configuraciones descritas, no obteniéndose ninguna dificultad.

Es éste Minigen un producto que además de ser reducido en su tamaño, lo cual siempre es interesante a la hora de calcular espacios de trabajo, también es reducido de precio.

Como se ha comentado al inicio de este Banco de Pruebas, el Minigen no es un paquete para uso profesional, estado este más indicado para un uso de nivel medio.

Los resultados obtenidos con el Genlock, dependerán en gran medida de la frecuencia de la señal. Si usa un video con sobresaturación de colores, por ejemplo, el Genlock ajustará la salida gráfica en concordancia con ello. Lo mismo ocurrirá con los sincronismos y niveles.

Dentro de los discos que componen el paquete MiniGen, se encuentra uno que contiene algunos dibujos IFF, así como un programa para emplear el modo interlazado.

En conjunto se puede hablar de este nuevo programa de Progressive Peripherals & Software como una buena aplicación para aquellos, que sin aplicaciones demasiado profesionales, deseen hacer sus propias digitalizaciones de video. ■

AMIGA ROM KERNEL REFERENCE MANUAL Y AMIGA HARDWARE REFERENCE

Distribuidor: ABC
Santa Cruz de Marcenado, 31
Precios: A consultar

Es muy probable que exista una gran cantidad de usuarios que necesiten, a la hora de crear un programa, acceder a una serie de librerías, así como conocer los requerimientos que estas funciones necesitasen. También es muy probable que muy pocos sepan cuáles son todas las librerías, así como la forma de acceder a las mismas.

Dichas dificultades, con las que se encuentran los programadores se deben a la falta de libros de referencia para dicho ordenador. Pues bien, en esta ocasión se comentarán dos libros indispensables en la biblioteca de cualquier programador de Amiga.

Amiga Rom Kernel

Para los poseedores de la antigua edición de este libro, este nombre no les resultará extraño. Esta nueva edición (revisada y actualizada), es como bien se dice en el sumario «la guía para todos aquellos programadores del Commodore Amiga». En esta nueva edición (de la Rom 1.3) se pueden encontrar los listados de las librerías, tanto en C como en ensamblador, usadas por la Rom. Ello es de gran utilidad a la hora de desarrollar sus propios programas (ya sea en lenguaje C o en lenguaje Ensamblador), puesto que encontrará todas las funciones, así como requisitos para acceder a las mismas.

Este libro se encuentra dividido en las siguientes secciones:

— Sumario para las funciones de las librerías del sistema.
— Sumario para los comandos de periféricos.
— Sumario para las llamadas del sistema.
— Ficheros Include en C.
— Ficheros Include en Ensamblador.
— Documentación sobre la «Amiga.lib» y código fuente.

— Documentación sobre el formato IFF.

Es sin lugar a dudas un libro indispensable para poder acceder a todas las funciones disponibles, así como para conseguir unos programas más fiables y de mayor velocidad en su ejecución.

Dentro del libro, el cual se compone únicamente de listados, se podrá encontrar todo lo referente al uso de los periféricos, procesadores, uso gráfico, manejo de ventanas, Request, Gadgets, etc. Así como una amplia información sobre el formato IFF. Es sin lugar a dudas un buen libro de información para saber, entre otras cosas, cómo funciona el ordenador.

Amiga Hardware Reference

El siguiente libro comentado es el Amiga hardware reference manual, el cual no se debe confundir con el Amiga hardware reference manual (Lybraries & Devices).

Este es un libro de GRAN UTILIDAD para comprender el manejo del Copper, sprites, pantallas, ventanas, creación de sampleados, control del sistema, Blitter, control del ratón etc.

El libro se encuentra dividido en ocho capítulos en los que todas estas funciones se encuentran perfectamente detalladas. Dichos capítulos son los siguientes:

— Introdución: dónde se hace una introducción al hardware Amiga.

— Hardware del coprocesador: dónde se explican las tres instrucciones utilizadas por el Copper, así como el manejo del mismo.

— Hardware de visualización: en este capítulo se puede encontrar todo lo referente a pantallas, ventanas, selección de colores, creacción de planos, etc.

— Hardware de los Sprite: como su nombre indica, en este capítulo se encuentra todo para el manejo de los Sprite.

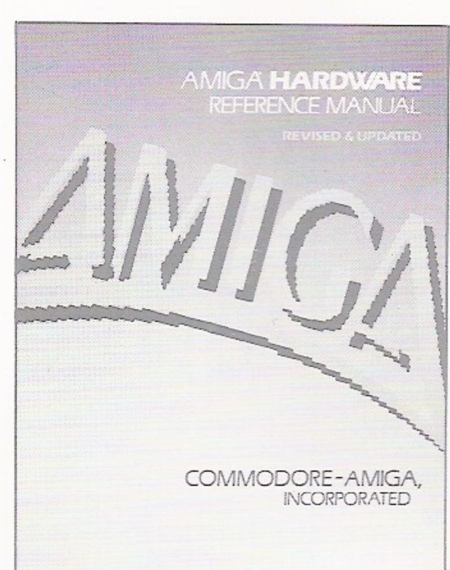
— Hardware Audio: en este capítulo, tras una introducción sobre el tema, se muestra la correcta utilización del hardware sonoro; explicando asimismo cambios de onda, selección de canales, etcétera.

— Hardware del Blitter: registros, multilaréa, etc.

— Hardware del sistema de control: interrupciones, prioridades de Sprites, colisiones, uso del interface.

— Hardware del Interface: registros del puerto de control, interrupciones de disco, cómo se transmiten los datos a los periféricos, etcetera.

Además de estos capítulos, el libro contiene los siguientes apendices:



Apendice A: Sumario de registros (en orden alfabético).

Apendice B: Sumario de registros (en orden de direcciones).

Apendice C: Lista de localizaciones del Custom chip.

Apendice D: Mapa de memoria del sistema.

Apendice E: Interfaces.

Apendice F: CIAs (puertos IúO, Modos de lectura, Timers, registro de control para interrupción de lectura, etcetera.

Apendice G: Tablas de autoconfiguración.

Apendice H: Comunicación del teclado, códigos de teclas, códigos especiales.

Apendice I: Conector de disco externo (spedificación del interface).

Seguramente serán muchos los usuarios, que con la ayuda de estos dos libros, consigan un mayor rendimiento de sus Amiga. ■



BCS—AMIGA—12

Este disco contiene gran cantidad de pequeños programas escritos en TRHU Basic. Dichos programas varían entre utilitarios, programas gráficos y de sonido. En muchos de los casos dichos programas son bastante simples, pero también útiles para los que acaban de introducirse en el mundo de la programación en el Amiga.

Los programas comprendidos son los siguientes:

Coins: Este es el típico juego creado con el comando RND, con el cual (a boleo) se decide cantidad de veces quién gana una partida.

Craps: Este juego (de sumar 15) está basado como el anterior en la instrucción RND.

Expense: Programa para representar datos mediante diagrama de barras.

Hanoi: Este programa es una demo basado en el conocido juego de las torres de Hanoi.

States: Este es un programa de preguntas y respuestas en el cual se deberá contestar correctamente las capitales de los Estados preguntados.

House: Este programa gráfico dibuja, con suma sencillez, casas con cualquier grado de inclinación.

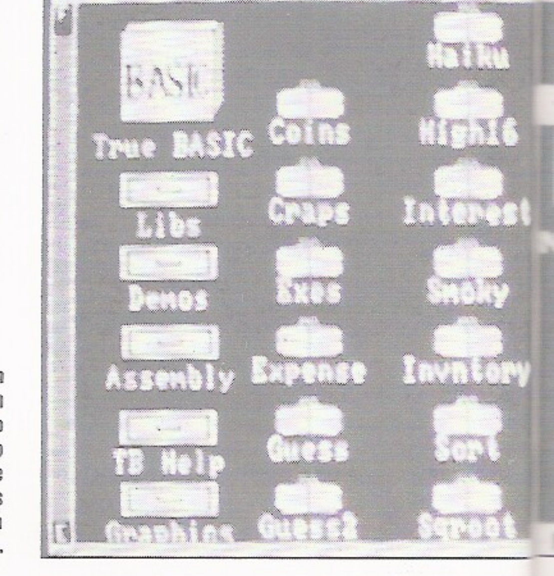
Además de estos programas, se presentan programas tales como una pequeña base de datos para llevar cualquier inventario. Aunque sencilla es útil y sencilla de manejar, además de permitir al usuario aprender más sobre el Amiga Basic.

También se presentan varias fuentes de programas en lenguaje Ensamblador; los cuales están listos para ensamblar y posteriormente linkar, permitiendo dejar éstos en formato ejecutable. PD #85

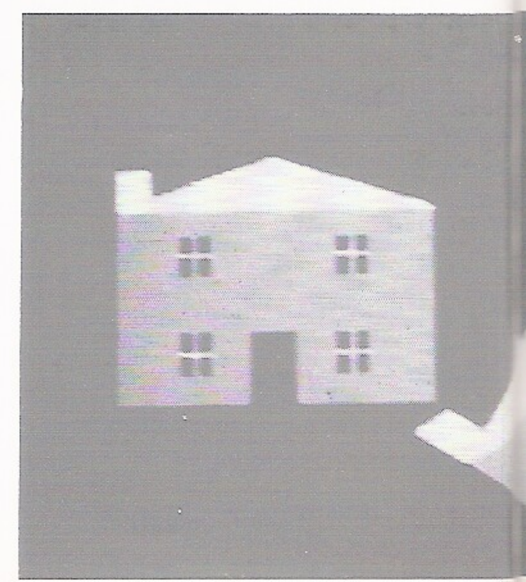
Lica—Amiga—24

La estructura de este disco de dominio público es la siguiente:

a) Utilidades.



Un buen disco lleno de programas para aprender.



Los discos de dominio público son gratis y a veces incluyen programas muy practicos que no se pueden encontrar en ningún otro sitio.

b) Juegos.
c) Arcs.

En el primero de los directorios (utilidades) se encuentra subdividido en dos más, entre los que destaca el directorio 3-D.

En este directorio se encuentran dos programas escritos en Basic. Uno de ellos (Rotation) es especialmente interesante, pudiendo observar cómo desde el Basic se pueden hacer buenas cosas; eso sí, no comparables a las realizadas en otros lenguajes.

Juegos

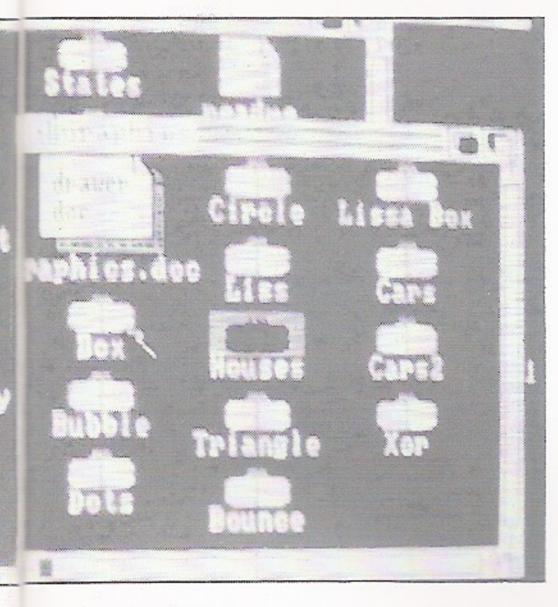
Dentro de este directorio se puede encontrar con el conocido juego de «los marcianos», sin lugar a dudas servirá mucho a la hora de divertirse entre utilitario y utilitario.

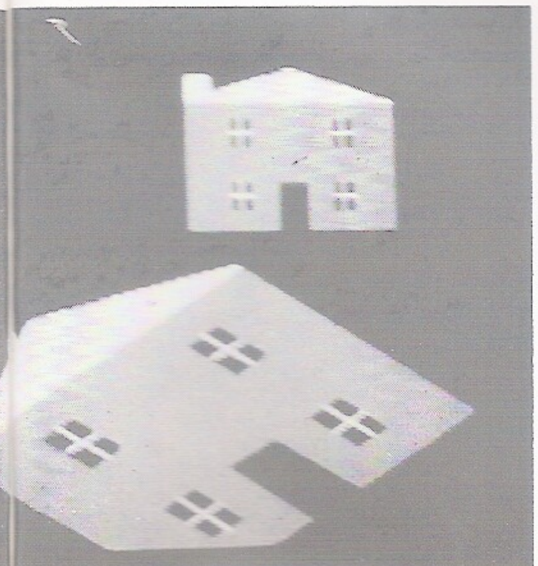
ARCS

Este es un directorio dedicado a poder incluir ficheros del tipo .ARC. Además este mismo posee un programa arqueador de inmensa utilidad a la hora de conseguir un mayor ahorro en el disco.

Entre los programas que contiene este disco en formato «arqueado» resalta uno de Cad 3-D, el cual invita a ser descomprimido para poder ver los resultados.

Otros programas son: 43 colores, ADDRIVER, DOS.COMMAND, DEBUG. PD 389 ■





Las curiosidades. así como las utilizades, se complementan...

COMMODORE

AMIGA

WORLD

N.º 4 - Publicación de IDG Communications - 500 ptas.

ESPECIAL JUEGOS

Guía, comentarios, trucos, ...

BANCO de PRUEBAS

A-MAX

Digi Paint 3

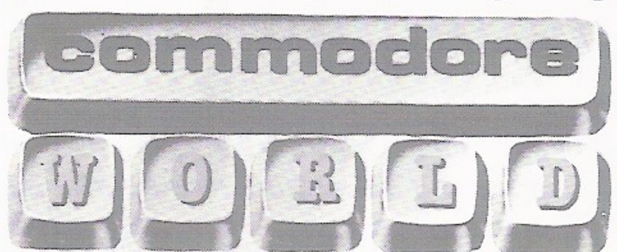
A 590

REPORTAJE

Centro del Diseño y la Comunicación



PEDIDOS



Para realizar sus pedidos copie o fotocopie esta página completa, marcando lo que desee con una cruz e indicando todos sus datos. No se atenderán pedidos con datos incompletos. Sume usted mismo el importe total del pedido y envíenos un cheque, giro o los datos de su tarjeta de crédito. Si paga por giro, indique también su pedido en el texto del resguardo.

NUMEROS DE LA REVISTA COMMODORE WORLD

Poner una cruz en los ejemplares que desees

Hasta el número 43 .... 300 ptas.

Del 44 al 64 .... 400 ptas.

Oferta Amiga en Commodore World (28, 34, 35, 38, 40 al 64). 4.500 ptas.

20	21	22	23	24	25	26	27	28
29	30	31	32	33	34	35	36	37
39	40	41	42	43	47	48	49	56
58	59	60	61	62	63	64		

EJEMPLARES ATRASADOS DE «CLUB COMMODORE» (Servicio de fotocopias)

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15

☐ Ejemplar Club Commodore .... 370 ptas.

☐ Oferta: Colección completa (16 números) .... 3.100 ptas.

BIBLIOTECA COMMODORE WORLD

☐ Volumen 1: Cursillo de código máquina .... 250 ptas.

☐ Volumen 2: Especial Utilidades .... 500 ptas.

☐ Disco Especial Utilidades .... 1.750 ptas.

☐ Oferta: Especial Utilidades + Disco .... 1.990 ptas.

PROGRAMOTECA COMMODORE WORLD

☐ Superdisco Aplicaciones I (dos discos) .... 1.990 ptas.

☐ Superdisco Aplicaciones II (dos discos) .... 1.990 ptas.

☐ Superdisco Aplicaciones III (dos discos) .... 1.990 ptas.

☐ Superdisco Aplicaciones IV (dos discos) .... 1.990 ptas.

☐ Superdisco Juegos .... 1.375 ptas.

Nombre y apellidos ....

Dirección ....

Población .... C.P. .... Provincia ....

Teléfono .... Modelo de Ordenador ....

Importe del pedido .... ptas.

Forma de pago: ☐ Cheque ☐ Giro número ....

☐ VISA ☐ MasterCard N.º ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

Fecha caducidad ☐☐ ☐☐ Firma

Los pedidos con tarjeta de crédito, sólo a partir de 3.000 ptas.
Gastos de envío e IVA incluidos.

NO SE ADMITEN PEDIDOS CONTRA REEMBOLSO

INTERFACE SERIE RS-

Por Gonzalo Fernánder Val.

En este artículo se realiza un interface, encargado de convertir las señales presentes en el port de usuario del C-64 en señales de niveles RS-232 y viceversa; para poder comunicar con otro ordenador que utilice el RS-232 estándar, como los Amiga o los compatibles PC.

El montaje lo realicé para poder sacar listados en Basic y código máquina por impresora, enviándolos a un PC primeramente y después del PC a la impresora. También se podría utilizar para hacer programas en código máquina, generar datos de sprites en un PC, mediante programas adecuados realizados en el PC (por ejemplo ensambladores cruzados), y mediante el interface, pasarlos a la memoria del C-64. Los programas en código máquina se pasarían a la memoria del C-64 mediante volcados hexadecimales.

Cada uno le puede dar la utilización que quiera a este interface, desde comunicarse a través de un modem hasta la utilización de periféricos tales como impresoras, ratones, etc., en el C-64.

Este interface es en realidad un cable de modem nulo junto con un conversor de voltajes. Es aplicable a cualquier modelo de la marca Commodore que tenga un port de usuario (VIC-20, Plus/4, C-64, C-128). Las pruebas con este interface principalmente las he realizado en un Plus/4 y también en un C-64. La parte de hardware es la misma para todos los ordenadores. Lo que cambia es la parte de software. Esto lo veremos más adelante.

Las normas RS-232 fueron definidas por la EIA (Electrical Industry Association) en cooperación con la Bell System, fabricantes de ordenadores y de modems con objeto de normalizar la conexión entre el equipo terminal de datos (DTE), como un ordenador, y el equipo de comunicación de datos (DCE), como un modem. Hoy día, se usa esta norma ampliamente para comunicarse ordenadores con periféricos tales como modems, plotters, impresoras, ratones, tabletas digitalizadoras, terminales de vídeo, etc., e incluso con otro ordenador, como explica este artículo.

Hardware

Para poder hacer las conversiones TTL-RS232 y RS232-TTL, se pueden utilizar varios sistemas. Afortunadamente, existen en el mercado dos circuitos integrados de interface. Estos son el 1488, que contiene 4 conversores TTL-RS232 (1 inversor y 3 puertas AND), y el 1489, con 4 conversores RS232-TTL (4 inversores trigger). Estos integrados son fabricados por diversas compañías, pero internamente son semejantes.

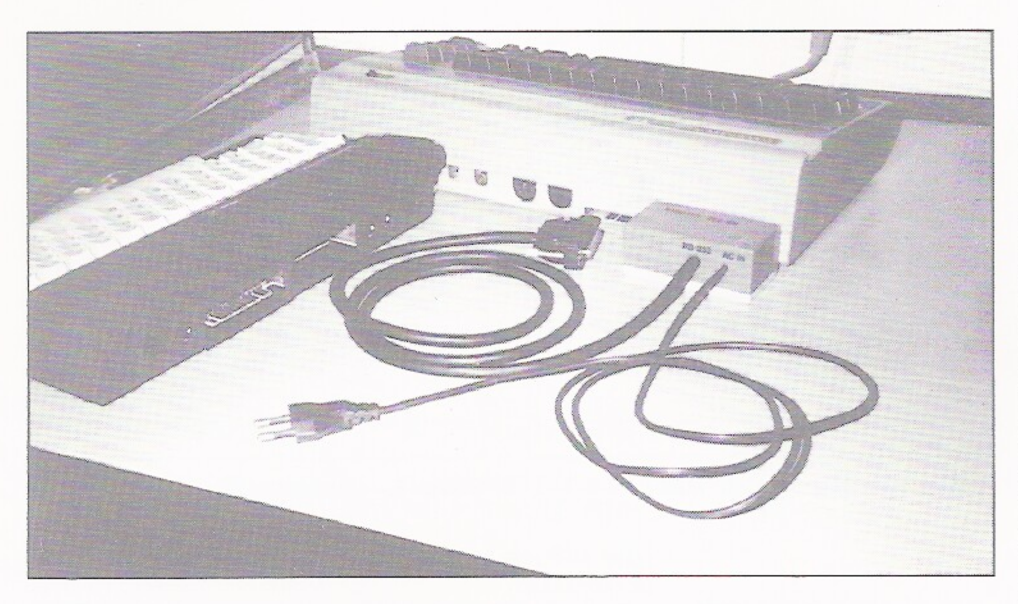
Los esquemas internos pueden verse en los dibujos 1 y 2.

En el 1488 se necesita una tensión de alimentación Vcc+= 9, 15V y Vcc-= -9, -15V con respecto al pin 7, que es la masa. En este montaje utilizamos Vcc+=12V y Vcc-=-12V a través de una pequeña fuente de alimentación. Para el 1489, se necesita una tensión Vcc= 5V. Dicha tensión la obtenemos del port de usuario.

El esquema del port de usuario (ver figura 1) cuando funciona como interface RS-232.

Lo que haremos será conectar una puerta del 1488 a cada línea de salida indicada por OUT en la fig. 1 y una puerta del 1489 a cada línea de entrada indicada por IN en la citada figura.

Es importante que las entradas de las puertas que no se utilicen en ambos integrados (1488 y



1489) se unan a masa (pin 7) para evitar un consumo de corriente innecesario.

Si queremos, podemos utilizar los pines B y C unidos, H, K, L conectados a un 1489 y los pines D, E y M conectados a un 1488 para realizar un verdadero interface RS-232, y así poder conectar el C-64 a un modem. Notemos que la línea RI (Ring Indicator) no se utiliza, debido a que de esta manera, utilizamos sólo dos integrados (1 1488 y 1 1489). No obstante, hay que tener en cuenta que algunos modems necesitan esta línea. Si realizamos el interface de esta manera, el cable de modem nulo necesario para poder conectarse con otro ordenador sería externo.

Los conectores RS-232 instalados en los PCs, Amiga, etc., son machos y de 9 o 25 pines, por lo que las salidas de los integrados 1488 y 1489 irán a conectores hembra presentes en el extremo del cable de modem nulo que utilice.

Las Líneas de dichos conectores más importantes son: PGND, TxD, RxD, RTS, CTS, DSR, GND, DCD, DTR Y RI.

Los esquemas de los conectores hembras son los siguientes (ver figura 2).

Mediante el cable de modem nulo, se trata de conectar dos equipos DTE. En este caso, hay que engañar a los dos equipos DTE haciéndoles creer que tienen conectado un equipo DCE. Esta función se realiza mediante el siguiente conexionado (existen otros esquemas posibles) (ver dibujo 3).

Las recomendaciones técnicas especifican que el cable une 2 DTE (2 ordenadores) o 1 DTE y 1 DCE (1 ordenador y 1 modem) no mida más de 15 metros de largo, aunque en la práctica, se pueden usar unos cuantos metros más. Mediante un bucle de corriente, las distancias entre los ordenadores puede ser hasta 1 Km. Para distancias más largas, se usa un modem acoplado a la red telefónica o una red local.

El esquema electrónico que he montado se indica en la fig. 2.

Se utilizan dos integrados reguladores de voltaje, uno para la alimentación positiva, que es el 7812, y otro para la alimentación negativa, que es el 7912. Los esquemas de ambos integrados son los siguientes (ver dibujos 4 y 5).

No es imprescindible el uso de una placa de circuito impreso para la realización del montaje. Puede hacer el montaje cableando todos los componentes. En este caso, para soldar un cable a una pata de los integrados, estañe antes el extremo del cable y la pata del integrado. De esta manera, será más fácil soldar los cables a los pines de los integrados. Este procedimiento puede ser empleado a todos los demás elementos. Cuidado con las equivocaciones. Haga las soldaduras lo más limpias posible y vigile el aislamiento entre las partes metálicas de los integrados reguladores (7812 y 7912) y los demás elementos. El transformador que yo he usado es uno de 9 + 9 voltios en alterna y baja potencia (2,5 VA), ya que el 1488 consume muy poca corriente si utilizamos una sola puerta y conectamos todas las entradas de las puertas que no se utilicen a masa, y me interesaba que fuera lo más pequeño posible para que pudiera meter todo el montaje dentro de una pequeña caja de plástico.

Puede utilizar un transformador de 12+12V y semejante potencia. Si va a realizar un interface RS-232 complejo, utilice un transformador de 12+12V 300mA para alimentar a 3 de las 4 puertas del 1488. Las entradas de la otra puerta conéctelas a masa, para que sea menor el consumo del integrado.

El cable que une el interface con el conector hembra de 25 o de 9 pines puede ser hasta 15 metros de largo, como ya he dicho. Es suficiente que sea de 3 conductores (lo mejor es 2 conductores + malla), ya que en la práctica el interface serie que lleva el PC y el Amiga, el pin 1 (PGND) es el mismo que el pin 7 (GND). En el caso del C-64 y del Plus/4 (y creo que también para los demás ordenadores Commodore), los pines A, 1, N y 12 son los mismos (GND). (Ver dibujo 6).

Recordemos que la alimentación del 1489 la obtenemos del port de usuario y que el consumo no puede ser mayor de 100 mA.

Puede acoplar un interruptor en serie con el primario del transformador y la red, y una luz de neón en los bornes del primario del tansformador como indicador de funcionamiento.

La lista de componentes es la siguiente:

3 condensadores 100 nF
2 condensadores 330 nF

S-232 PARA C-64

2 condensadores 47 uF/25V
2 condensadores 1000 uF/25V
1 condensador 10 uF/25V
1 puente B80-C1000
1 IC 7812
1 IC 7912
1 IC MC1488
1 IC MC1489
1 transformador 9+9V 2,5VA
1 conector hembra de 25 o de 9 pines
1 conector para el port de usuario
1 caja de plástico o de metal
Opcionalmente, un interruptor y una luz de neón.

Software

El chip que gobierna el port RS-232 en el C-64 es una CIA (Adaptador de Interface Complejo) 6526 de las dos con que va equipado. En el PLUS/4 es el ACIA 6551 (Interface Adaptador de Comunicaciones Asíncronas). Ambos chips son muy parecidos en lo relativo a las comunicaciones asíncronas RS-232.

Estos chips constan de una serie de registros internos encargados de realizar y supervisar la comunicación serie. Los registros más importantes son el registro de control, el registro de comando y el registro de estado.

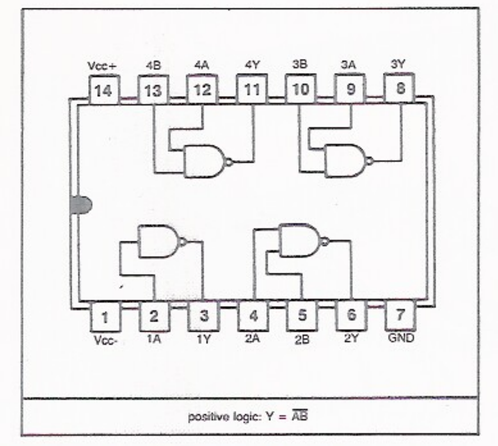
Estos registros pertenecen al chip 6551. En el chip 6526, el bit 4 del registro de control y los bits 1, 2 y 3 del registro de comando (registro de mando) no se utilizan. El valor del registro de estado se guarda en la variable ST, accesible desde el BASIC.

Para poder comunicar con un PC o Amiga, se necesita un programa de comunicaciones en el C-64 y otro en el PC/Amiga.

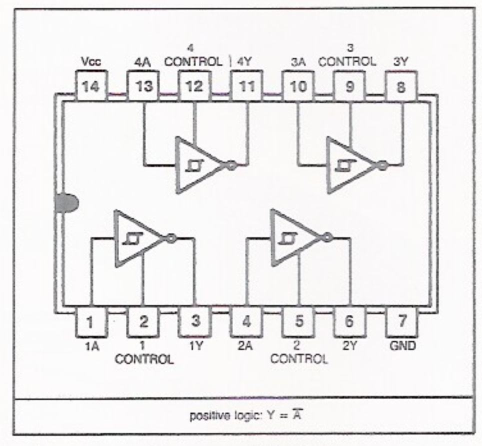
Yo he utilizado para el PC un programa de comunicaciones conocido: ProCom Versión 2.4.2, de DATASTORM Technologies.

Para el Amiga se puede utilizar RUNTERM, programa publicado por Commodore World en su disco Aplicaciones IV. Este programa dispone de dos versiones, C-64 y C-128, está perfectamente probado con PC y Amiga, además es muy completo.

Una vez conectado el interface al C-64/Plus/4 y al PC/Amiga a través del conector hembra que lleva el cable de modem nulo, hay que configurar los parámetros de comunicaciones para ambos ordenadores.



Dibujo 1.— Esquema MC 1488



Dibujo 2.— Esquema MC 1489

En el ProComm, el proceso de configuración es el siguiente:

Una vez cargado ProComm, pulsar Alt-S para acceder al menú de instalación.

— Pulsar la opción 2 (TERMINAL SETUP).

— Pulsar la opción 4 y dejarla en CR/LF.

— Pulsar la opción 5 y dejarla en CR.

— Pulsar ESC para salir del menú TERMINAL SETUP y volver al menú de instalación.

— Pulsar la opción 6 (ASCII TRANSFER SETUP).

— Pulsar la opción 1 y dejarla en eco local.

— Pulsar la opción 8 y dejarla en ADD LF.

— Pulsar ESC para volver a la instalación.

— Si se quiere, se pueden salvar los cambios hechos pulsando la opción S).

— Pulsar ESC para volver a la pantalla TERMINAL.

— Pulsar Alt-P para seleccionar el puerto al que está conectado el interface, (COM 1, COM 2, ...), la velocidad en baudios, la paridad, el número de bits de datos, y el nº de bits de stop. Se pueden salvar los cambios hechos por la opción 24.

> **Para poder comunicar con un PC a Amiga, se necesita un programa de comunicaciones en el C-64 y otro en el PC/Amiga.**

En comunicaciones, las selecciones más comunes son N/8/1 (sin paridad, 8 bits de datos y 1 bit de stop) y E/7/1 (paridad par, 7 bits de datos y 1 bit de stop). Muchas bases de datos (BBS) requieren N/8/1 y muchos grandes ordenadores utilizan el formato E/7/1. Yo he utilizado el formato N/8/1.

Una vez hechos los anteriores cambios en el programa ProComm, el programa de comunicaciones para el PC ya está configurado.

La sentencia para poder comunicar desde el C-64 con el interface RS-232 es la siguiente:

OPEN lfn,2,0,CHR$(registro de control)+CHR$(registro de comando)

lfn es el número lógico de fichero. Puede ir desde 1 hasta 255. Si se escoge un número superior a 127, entonces se envía un carácter LF a continuación de cada carácter CR enviado. Nosotros escogeremos un número menor que 128, pues el programa ProComm ya está configurado para que añada un caracter LF a cada caracter CR que reciba.

2 es el número de periférico, que corresponde al RS-232.

0 indica que no hay dirección secundaria.

El registro de control y el registro de comando son 2 números decimales obtenidos a partir de la fig. 3, asignando un 1 ó 0 a cada bit de cada registro y calculando el valor decimal correspondiente a el grupo de 8 bits pertenecientes a cada registro.

Comunicación C-64 → PC

Hay que tener en cuenta que en el C-64 se pueden enviar datos al PC a una velocidad máxima de

2400 baudios, y en el Plus/4 a 4800 baudios. Si los listados en BASIC, código máquina a enviar son cortos, dichos listados se pueden enviar desde el Plus/4 a una velocidad de 19200 baudios. Si son largos (unas 300 líneas de volcado hexadecimal a través del monitor incorporado), puede haber desincronización y salen líneas mal posicionadas. Lo mismo pasa con velocidades superiores a 4800 baudios.

Por ejemplo, para enviar un listado en BASIC del C-64 al PC, a 2400 baudios, tecleamos en el C-64:

```
OPEN 2,2,0,CHR$(10)+CHR$(5)
```

En el PC, con el programa ProComm cargado, pulsamos la tecla PgDn (Download), seleccionamos protocolo ASCII, y le damos un nombre de fichero a los datos que obtengamos del C-64.

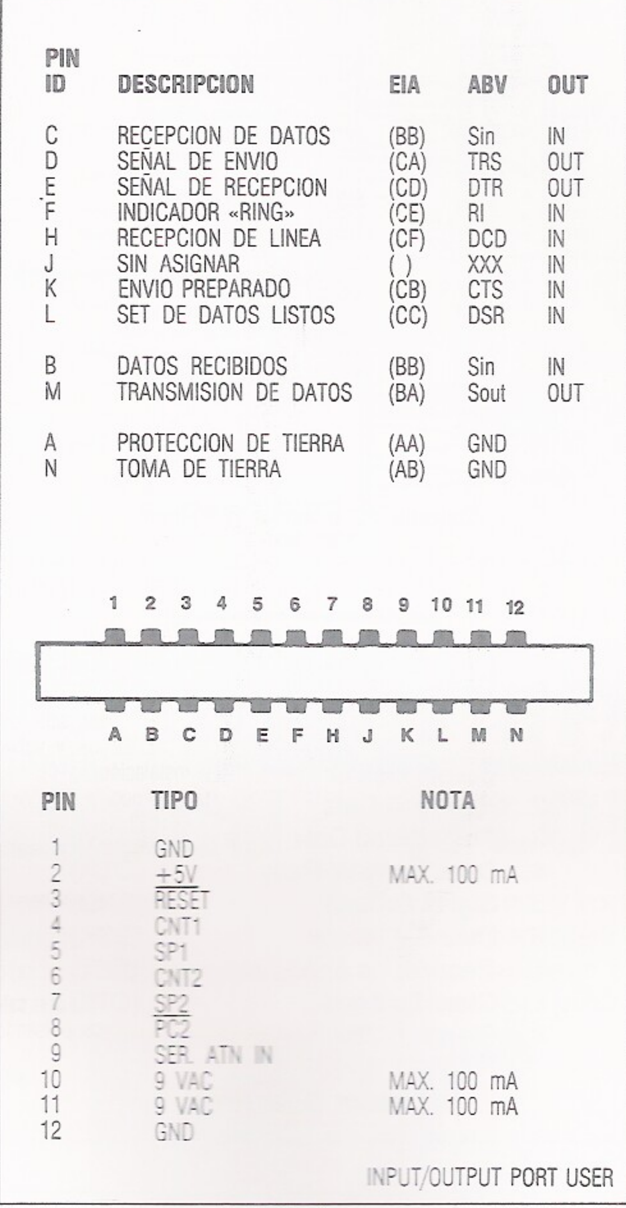
En el C-64, tecleamos CMD 2. La palabra «READY» debe salir en la pantalla del PC. A continuación, tecleamos LIST y nos saca el listado BASIC por la pantalla del PC.

Para acabar de enviar datos al PC, tecleamos PRINT#2 para redireccionar la salida a la pantalla del C-64 en lugar del interface RS-232 y cerramos el fichero abierto con CLOSE 2.

La anterior sentencia OPEN es válida tanto para el C-64 como para el Plus/4.

Si al enviar datos del C-64 al PC, éstos nos salen ininteligibles, es posible que los parámetros de comunicación en el PC (accesibles mediante ALT-P, en el programa ProComm) no sean iguales a los del C-64 (distintas velocidades de transmisión en el C-64 y recepción en el PC, número distinto de bits de stop,...).

Tengamos en cuenta que el C-64 utiliza un código ASCI no standard, por lo que nos pueden salir signos raros en la pantalla del PC si enviamos listados en BASIC que contienen caracteres gráficos. Para solventar este problema, el programa ProComm tiene una tabla de traslación accesible por Alt-W que lo que hace es una conversión de los caracteres recibidos a caracteres que nosotros definimos basados en el código ASCII del PC.



PIN ID	DESCRIPCION	EIA	ABV	OUT
C	RECEPCION DE DATOS	(BB)	Sin	IN
D	SEÑAL DE ENVIO	(CA)	TRS	OUT
E	SEÑAL DE RECEPCION	(CD)	DTR	OUT
F	INDICADOR «RING»	(CE)	RI	IN
H	RECEPCION DE LINEA	(CF)	DCD	IN
J	SIN ASIGNAR	( )	XXX	IN
K	ENVIO PREPARADO	(CB)	CTS	IN
L	SET DE DATOS LISTOS	(CC)	DSR	IN
B	DATOS RECIBIDOS	(BB)	Sin	IN
M	TRANSMISION DE DATOS	(BA)	Sout	OUT
A	PROTECCION DE TIERRA	(AA)	GND	
N	TOMA DE TIERRA	(AB)	GND	

PIN	TIPO	NOTA
1	GND	
2	+5V	MAX. 100 mA
3	RESET	
4	CNT1	
5	SP1	
6	CNT2	
7	SP2	
8	PC2	
9	SER. ATN IN	
10	9 VAC	MAX. 100 mA
11	9 VAC	MAX. 100 mA
12	GND	

INPUT/OUTPUT PORT USER

Fig. 1.—Líneas del Port de Usuario perteneciente al ordenador.

Comunicación PC → C-64

Para poder enviar ficheros del Pc al C-64 primeramente hemos de saber cómo se compone un fichero ASCII creado por cualquier procesador de texto en un PC.

Un fichero de texto en un PC, tiene el aspecto siguiente:

```
[cadena de caracteres] [CR] [LF]
                    CR = CHR$ (13)
[cadena de caracteres] [CR] [LF]
                    LF = CHR$ (10)
[cadena de caracteres] [CR] [LF
                   EOF= CHR$ (26)
[cadena de caracteres] [CR] [LF]
[EOF] (finales de fichero)
```

Al enviar el fichero de texto el PC al C-64, el C-64 ha de saber cúando termina el fichero. Esto se podría hacer chequeando si se envía desde el PC un carácter EOF (CHR$(26)). Lo que pasa, es que la indicación de que se ha terminado de enviar un fichero ASCII a través del programa ProComm se realiza por las líneas de comunicación, por lo que nunca se envía el carácter EOF al C-64, y el programa receptor en el C-64 siempre estaría a la espera de recibir datos del PC. Una manera fácil de solventar este inconveniente es teclear mediante un procesador de texto un carácter llamado centinela al final del fichero que queremos enviar. Este podría ser el Alt 255 (manteniendo pulsada la tecla Alt. tecleamos 255 en el teclado numérico de la derecha del teclado). Por consiguiente, la estructura del fichero de texto a enviar será ahora la siguiente:

```
[cadena de caracteres] [CR] [LF]
[cadena de caracteres] [CR] [LF]
[cadena de caracteres] [CR] [LF]
[cadena de caracteres] [CR] [LF]
[cadena de caracteres] [CR] [LF]
Alt 255 (espacio en blanco)
[EOF]
```

Hemos de utilizar técnicas de teclado dinámico explicadas en el nº 33 de Commodore World si queremos introducir en la memoria del C-64 el programa BASIC recibido, suponiendo que el fichero enviado desde el PC sea de un programa en BASIC.

Supongamos que queremos enviar el siguiente programa en BASIC al C-64:

```
10 PRINT «ESTO ES UNA PRUEBA»
20 END
```

El anterior programa lo creamos mediante un procesador de texto, o más rápidamente COPY CON PRUEBA.BAS.

No olvidemos de dejar pulsada la tecla CAPS LOCK para que las letras salgan en mayúsculas, ya que las letras minúsculas tienen un código ASCII distinto de las letras mayúsculas.

Sin olvidar el carácter centinela Alt 255, el fichero PRUEBA.BAS tendrá la siguiente estructura:

```
10 PRINT «ESTO ES UNA PRUEBA»
[CR] [LF]
20 END [CR] [LF]
Alt 255
[EOF]
```

El carácter LF lo introduce el procesador de texto después de cada carácter CR pulsado, por lo que no tienes que preocuparte de introducirlo.

Evitemos crear líneas de más de 80 caracteres, pues no estaría en la memoria del C-64 al posicionar el cursor sobre dicha línea y pulsar RETURN.

Ahora cargamos el programa ProComm en el PC y seguidamente tecleamos el listado 1 en BASIC en el C-64 o Plus/4.

La explicación del programa es la siguiente:

Línea 0: Abrimos el canal RS-232 para una recepción de 300 baudios.

Línea 1: Posicionamos el cursor en la línea BASIC recibida.

Línea 2: Obtenemos un carácter del buffer RS-232, que contiene los caracteres enviados por el PC.

Línea 3: Obtenemos el valor decimal asociado al carácter ASCII CS. Si vale 255 (fin de fichero) terminamos.

Línea 4: Si el carácter recibido es un RETURN, simulamos la pulsación de 2 RETURN, uno para entrar en la memoria del C-64 la línea recibida, y otro para volver a ejecutar el programa.

Línea 5: El carácter recibido no es un RETURN ni fin de fichero (Alt-255), por lo que lo imprimimos y volvemos a recibir más caracteres.

Tecleamos la siguiente línea en modo directo para cambiar el color de fondo, el color del borde y el color del cursor en el C- 64:

POKE 53280,0:POKE 53281,1:PRINT «[BLK]»

Ahora ejecutamos el programa con RUN. Veremos que la pantalla se queda en blanco. Seguidamente, pulsamos la tecla PgUp (Upload) en el PC, seleccionamos protocolo ASCII y tecleamos el fichero a enviar, que en nuestro caso es PRUEBA.BAS. Al pulsar RETURN veremos en la pantalla del C-64 cómo van saliendo carácter por carácter las dos líneas de que consta el programa PRUEBA.BAS. Si hemos tecleado el carácter Alt 255 al final del fichero PRUEBA.BAS, el programa en el C-64 eventualmente termina, y al hacer en LIST, nos aparecen las líneas del programa receptor tecleando anteriormente y las dos líneas que componen el programa PRUEBA.BAS.

-Si surgen problemas, vigile que cada línea enviada desde el PC al C-64 no tenga más de 80 caracteres. También es válido lo dicho cuando surgen problemas al enviar desde el C-64 al PC.

Este programa BASIC funciona sólo a 300 baudios por segundo tanto en el C-64 como en el Plus/4. Si seleccionamos una velocidad más alta en cualquiera de los dos ordenadores, ya no hay seguridad en los datos recibidos y las líneas salen mal formadas. También influye que el cable de modem nulo tenga buen aislamiento y esté lejos de fuentes parásitas de ruido, como transformadores, motores, etc. A veces, en el C-64 y en el Plus/4 se pueden recibir programas de 2400 baudios si son cortos y el cable está aislado convenientemente.

Para poder recibir datos a mayor velocidad, es necesario usar el código máquina. Hay dos rutinas, una para C-64 y otra para el Plus/4 que permiten recibir datos con seguridad a 300 y 1200 baudios respectivamente. La diferencia entre estas dos velocidades de recepción creo que se encuentra en que ambos ordenadores usan chips distintos para comunicaciones (una CIA 6526 en el C-64 y una ACIA 6551 en el Plus/4). o obstante, es probable que con esta rutina puedas recibir en el C-64 programas a 2400 baudios sin errores que con la anterior en BASIC. Los valores por defecto que llevan estas dos rutinas son recepción a 300 baudios en el C-64 y recepción a 1200 baudios en el Plus/4. Si quieres cambiar estos valores, has de teclear:

POKE 49249, (registro de control) en el C-64 y
POKE 1642, (registro de control) en el Plus/4

Recuerde que el bit 4 del registro de control ha de valer 1 para el Plus/4. Los anteriores POKEs se entran después de ejecutar las mencionadas rutinas en código máquina. Veremos que al ejecutar

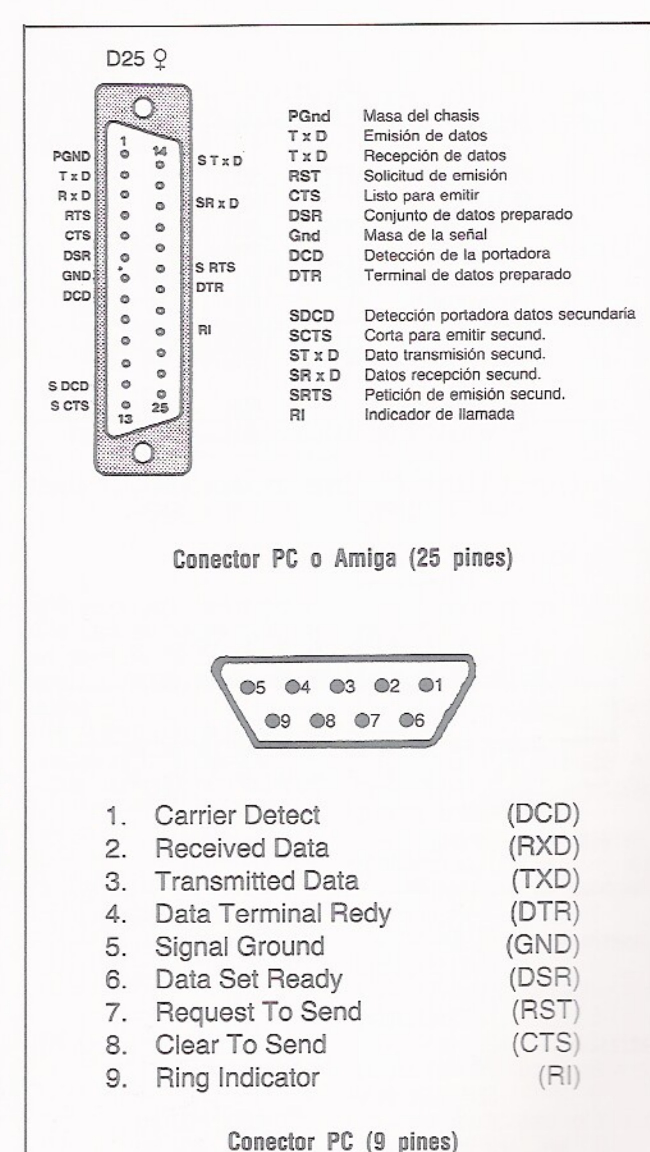
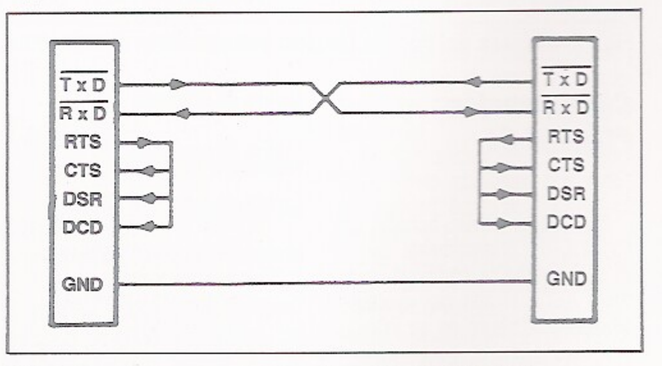


Fig. 2.

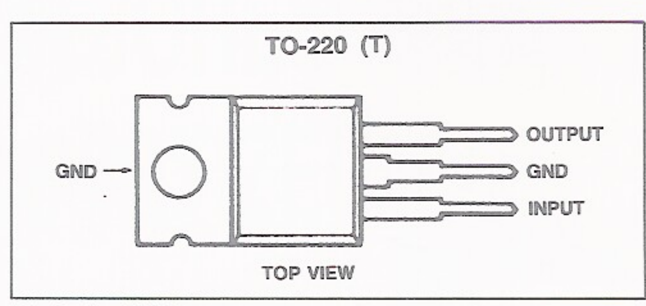


Dibujo 3.—Esquema conexionado cable Modem nulo.

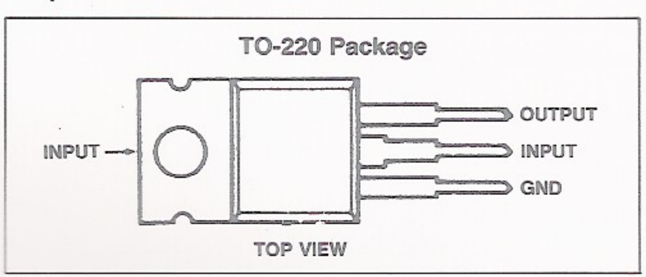
la rutina en código máquina, salen dos opciones: recibir programas BASIC y recibir programas C.M.

La segunda opción sólo se puede usar en el C-64 si se tiene conectado algún cartucho monitor de código máquina. Es probable que tal como está la rutina para C-64 no puedas recibir volcados hexadecimales (aparecen con el comando M del monitor) de una manera satisfactoria. En este caso se tienen que hacer cambios en el programa de acuerdo con la sintaxis de una línea de volcado hexadecimal. La sintaxis que manejan las dos rutinas (para el Plus/4 y para el C-64) es la de los monitores de código máquina incorporados en el C-16, Plus/4 y C-128.

Las líneas de volcados hexadecimales creadas por los procesadores de texto en los PCs tienen que tener el siguiente formato:



Dibujo 4.—7812.



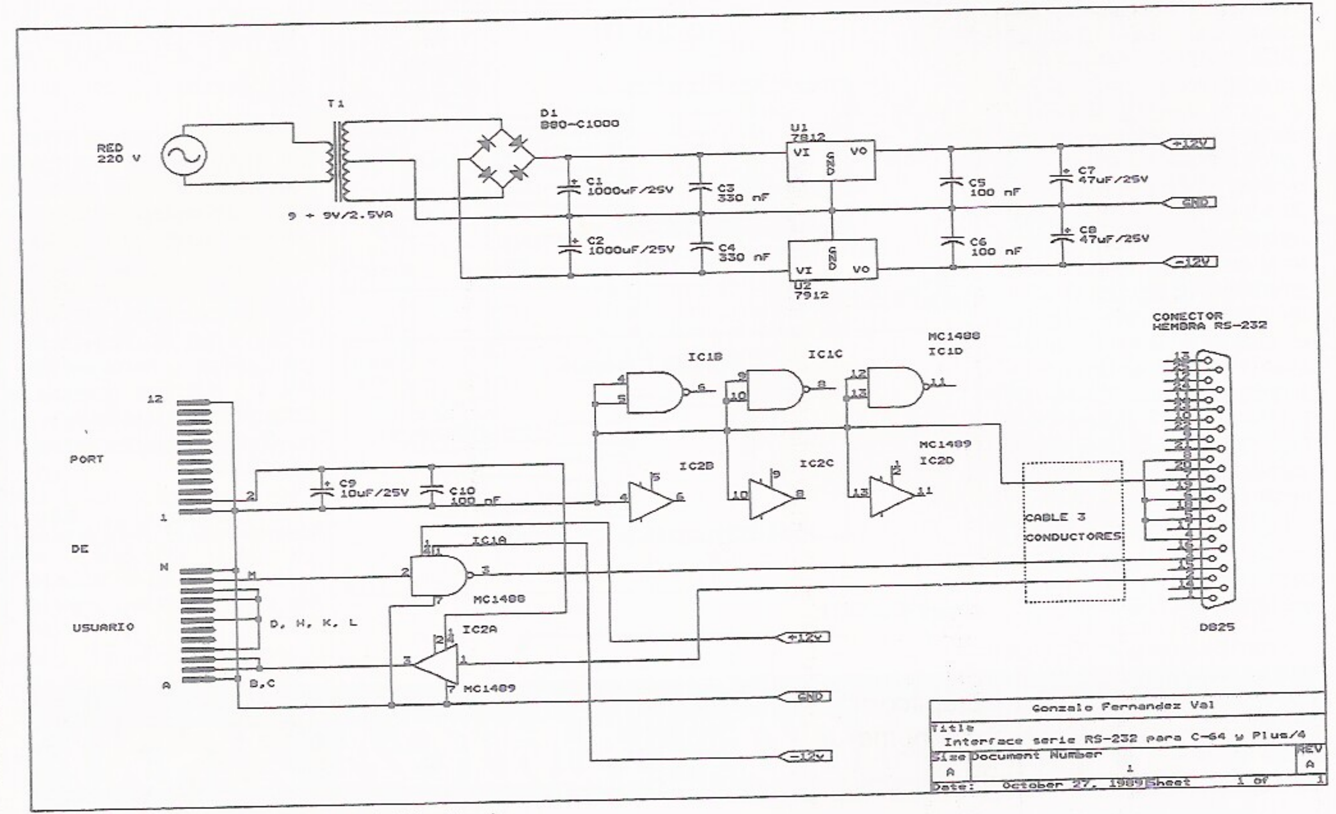
Dibujo 5.—7912.

```
Ejemplo:]C0AA AA 10 30 37 89 8E CR LF
]C0B2 12 13 16 34 65 76 43 78 CR LF
Alt 255
[EOF]
```

Como vemos, han de empezar con el carácter], tal como aparecen al visualizarlas con el comando M del monitor de código máquina incorporado en el Plus/4 y C-128.

La adaptación de estas rutinas a los demás ordenadores (VIC- 20 y C-128) es muy fácil, ya que el programa utiliza las rutinas del KERNAL y 3 posiciones de memoria correspondientes al buffer de teclado. Son las siguientes: (tomadas del nº 33 de CW).

Ordenador	Número de caracteres	Buffer de teclado
VIC y C-64	198	631-640
Plus/4, C-16	239	1319-1328
C-128	208	832-851



Dibujo 6.—Interface RS-232 para C-64 y Plus/4.

```
PROGRAMA: RS-2.1                        LISTADO 1

0 OPEN 2,2,0,CHR$(22)+CHR$(5):REM E .138
L BIT 4 DE CONTROL VALE 1
1 PRINT "[BLK][HOM][3CRSRD]";         .243
2 GET#2,C$:IFC$="" THEN 2             .174
3 C=ASC(C$):IF C=255 THEN END:EL EN .37
D CIERRA TODOS LOS FICHEROS
4 IF C=13 THEN PRINT C$"[WHT][CTRL  .18
C][HOM]":POKE 198,2:POKE 631,13:POK
E 632,13:END
5 REM POKE 239,2:POKE 1319,13:POKE  .223
1320,13 PARA EL PLUS/4
6 PRINT C$;:GOTO 2                    .180
```

```
PROGRAMA: RS-2.2                        LISTADO 2

0 POKE 53280,0:POKE 53281,1:PRINT " .110
[BLK]"
1 PRINT"[CLR]1) RECIBIR PROGRAMAS B .139
ASIC":PRINT"2) RECIBIR PROGRAMAS C.
M.":INPUT"OPCION";O
2 FOR A=49152 TO 49260:READ D:POKE  .60
A,D:S=S+D:NEXT
3 READ D:IF D<>S THEN PRINT "[CLR]E .73
RROR EN DATAS!":END
4 IF O=1 THEN 7                       .132
5 POKE 49186,3:POKE 49221,0:POKE 49 .139
248,0
6 FOR A=49252 TO 49259:READ D:POKE  .0
A,D:NEXT
7 PRINT "[CLR]TECLEE SYS49152 PARA  .149
RECIBIR.":END
8 DATA 169,144,32,210,255,169,2,162 .140
,2,160,255,32,186,255,169,2,162,97,
160,192
9 DATA 32,189,255,32,192,255,169,14 .249
7,32,210,255,169,13,162,1,32,210,25
5,202,16
10 DATA 250,162,2,32,198,255,32,228 .80
,255,201,0,240,249,201,255,240,9,32
,210,255
11 DATA 201,13,240,6,208,236,32,231 .145
,255,96,162,9,189,99,192,32,210,255
,202,16
12 DATA 247,169,2,133,198,169,13,14 .250
1,119,2,141,120,2,32,231,255,96,22,
5,19,50
13 DATA 53,49,57,52,83,89,83,5,1447 .177
0,32,32,32,48,48,48,67,71
```

```
PROGRAMA: RS-2.2                        LISTADO 3

0 POKE 53280,0:POKE 53281,1:PRINT " .110
[BLK]"
1 PRINT"[CLR]1) RECIBIR PROGRAMAS B .139
ASIC":PRINT"2) RECIBIR PROGRAMAS C.
M.":INPUT"OPCION";O
2 FOR A=49152 TO 49260:READ D:POKE  .60
A,D:S=S+D:NEXT
3 READ D:IF D<>S THEN PRINT "[CLR]E .73
RROR EN DATAS!":END
4 IF O=1 THEN 7                       .132
5 POKE 49186,3:POKE 49221,0:POKE 49 .139
248,0
6 FOR A=49252 TO 49259:READ D:POKE  .0
A,D:NEXT
7 PRINT "[CLR]TECLEE SYS49152 PARA  .149
RECIBIR.":END
8 DATA 169,144,32,210,255,169,2,162 .140
,2,160,255,32,186,255,169,2,162,97,
160,192
9 DATA 32,189,255,32,192,255,169,14 .249
7,32,210,255,169,13,162,1,32,210,25
5,202,16
10 DATA 250,162,2,32,198,255,32,228 .80
,255,201,0,240,249,201,255,240,9,32
,210,255
11 DATA 201,13,240,6,208,236,32,231 .145
,255,96,162,9,189,99,192,32,210,255
,202,16
12 DATA 247,169,2,133,198,169,13,14 .250
1,119,2,141,120,2,32,231,255,96,22,
5,19,50
13 DATA 53,49,57,52,83,89,83,5,1447 .177
0,32,32,32,48,48,48,67,71
```

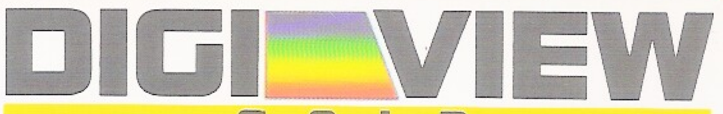
La rutina permanece en la memoria después de hacer un RESET en el C-64 o en el Plus/4, con lo que sólo hay que cargarla una vez.

La rutina se puede colocar en cualquier lugar de la memoria, siempre que no interfiera con otros programas o con posiciones de memoria reservadas.

Estas rutinas se pueden ampliar para que incluyan filtros de caracteres, para poder pasar del ASCII de los PCs al ASCII de Commodore. El lector puede consultar los números 10, 33, 36, 45, 49 y 51 de Commodore World para obtener más información de cómo ampliar la rutina y personalizarla a gusto de cada uno.

Finalmente, para el C-64, lo más fácil sería utilizar un paquete de comunicaciones, como el comentado en el número 62 de Commodore World, VIP TERMINAL.

Esperamos que este artículo represente una ayuda importante a los usuarios, muchos de los cuales disponen de varios ordenadores a la vez.



¡NUEVO!
Hardware y Software
Diseñado para A500/2000
¡Incluso mejores imágenes que antes!



1.



2.



Imagen en 4096 colores creada con Digi-View Gold

3. Simplemente el mejor.

El nuevo Digi-View Gold es el mejor digitalizador de imágenes para el Amiga. Ningún otro se le parece. ¿Por qué? El secreto es que Digi-View Gold captura 2,1 millones de colores en memoria, creando la increíble sensación de 100.000 colores en pantalla simultáneamente.

Y es sencillo de usar. Simplemente enfoque su cámara de vídeo sobre cualquier objeto o fotografía, y en segundos Digi-View Gold lo convertirá en un gráfico de Amiga llena de colores vibrantes y de claridad. Si usted está creando gráficos para autoedición, presentaciones, vídeo, o sólo por diversión, Digi-View Gold crea brillantes imágenes con asombrosa facilidad.

Digi-View está específicamente diseñado para el Amiga 500 y 2000, y se conecta directamente en el puerto paralelo de su ordenador. El potente software de captura y manipulación de imágenes (versión 3.0) incluido en su Digi-View Gold tiene un completo control del color y el sombreado, modo overscan PAL, extra halfbrite y el modo especial «Line Art» para autoedición.

Solamente Digi-View Gold:

- Puede digitalizar en todos los modos de resolución del Amiga desde 320 × 256 PAL hasta 768 × 580 (alta resolución PAL overscan).
- Maneja desde 2 hasta 4096 colores (incluido extra halfbrite).
- Soporta el exclusivo HAM potenciado para resaltar los más pequeños detalles.
- Es 100% compatible IFF y trabaja con cualquier programa de dibujo.
- Puede digitalizar hasta 21 bits por pixel (2,1 millones de colores) para obtener las mejores imágenes posibles.
- Incorpora avanzadas rutinas de optimización que dan la apariencia de 100.000 colores en pantalla simultáneamente.
- Posee un potente sistema de procesado de imágenes que permite la completa manipulación de pantalla IFF.

Si usted desea la mayor calidad en sus gráficos de Amiga, y tan fáciles de obtener como contar hasta tres, entonces usted necesita la nueva versión del digitalizador de imágenes más vendido de todos los tiempos: Digi-View Gold.

Cada Digi-View Gold incluye hardware de digitalización, rueda de filtros, software de digitalización versión 3.0 (próximamente en castellano), garantía ilimitada y manual de instrucciones en castellano.

SOLO POR
25.900 ptas.
+ 12% IVA



DISTRIBUIDOR EXCLUSIVO
PARA ESPAÑA
DE NEWTEK INC.



PLAZA ISABEL LA CATOLICA, N.º 1 TELF. (988) 75 11 80; FAX: (988) 75 11 91 34005 PALENCIA